Low-Density Parity-Check Codes with Erasures and Puncturing

A Thesis

Presented to

The Academic Faculty

By

Jeongseok Ha

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy

School of Electrical and Computer Engineering
Georgia Institute of Technology
November 2003

Low-Density Parity-Check Codes with Erasures and Puncturing

Approved by:

Steven W. McLaughlin, Advisor

John R. Barry

Ye (Geoffrey) Li

Date Approved November 17, 2003

ACKNOWLEDGEMENTS

I am grateful to my advisor, Dr. Steven W. McLaughlin for his advice. He has always guided me to new research topics and watched me with patience. I would like to express my gratitude for his cares not only for my research but also for my family. I have really enjoyed discussions with him. This dissertation would not have been possible without his guidance.

I also thank the members of my dissertation committee, Dr. John R. Barry, Dr. Ye (Geoffrey) Li, Dr. Hsien-Hsin Sean Lee and Dr. Prasad Tetali for serving on the committee. Their valuable comments have been immensely useful in helping me make this dissertation improved.

I would like to thank all my fellow graduate students for their help. Especially, I should thank Deric Waters and Woojay Jeon for their proof-reading of dissertation and papers. Thanks also go to Jeahong Kim, Woonhaing Hur, Soo-jung Ryu, Jungwon Kang and Kasyapa Balemarthy for their friendly conversations and help.

Finally, I would like to thank to my family. I thank my mother for her love and help during my entire life. After being a father, now I can barely understand her unmeasurable love. My wife, Mijeong has been a great supporter since the first day at Georgia Tech. Her support, patience and love make this dissertation possible. My baby, Brian gives me joy that I have not experienced. I thank my wife for taking care of him and scarifying her life for family.

Although I forget the names, there must be a lot of persons who have helped me. I thank you all!

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SUMMARY

Many research results have shown that well-designed low-density parity-check (LDPC) codes can achieve capacity-approaching performances over important channels such as additive white Gaussian noise, binary-erasure, and binary-symmetric channels. However, over the channels, all coded symbols in a codeword are statistically equally corrupted. In this thesis, we think of systems in which coded symbols of a codeword are divided into subgroups and the symbols in the subgroups are transmitted through their corresponding sub-channels.

In magnetic storages, coded symbols are either corrupted by Gaussian noise or erased due to thermal asperity, which can be modelled as a combination/mixture of additive-white Gaussian-noise (AWGN) channel and binary erasure channel (BEC) [16, 18]. In holographic storages, signal-to-noise ratios (SNR) of coded symbols depend on the areas where the symbols are recorded [35]. We can model the holographic storages as a combination of sub-channels with different SNRs. In rate-compatible punctured codes, a portion of symbols of a codeword is deliberately deleted in the transmitter to increase coding rate and the remaining coded symbols are transmitted. Thus, we can model punctured codes as a combination of two sub-channels and two subgroups of coded symbols [17, 19, 20].

We analyze behaviors of LDPC codes with the message-passing decoders in the proposed systems. There are two well-known techniques for analyzing LDPC codes, which are density evolution and Gaussian approximation techniques. Both techniques trace variations of probability densities of messages in the message-passing decoders during iterations. The former heavily depends on numerical analysis and makes accurate results at the sacrifice of analytical insight. Gaussian approximation models the

probability densities as Gaussian and gives us a closed form of a recursive equation describing the variation of the probability densities during iterations.

First, we consider LDPC code design for AWGN channels with erasures. This model, for example, represents a common situation in magnetic and optical recording where defects or thermal asperities in the system are detected and presented to the decoder as erasures. We give thresholds of regular and irregular LDPC codes and discuss practical code design over the mixed Gaussian/erasures channel. The analysis is an extension of the Gaussian approximation work of Chung *et al* [8]. In the two limiting cases of no erasures and large SNR, the analysis tends to the results of Chung *et al.* [8] and Luby *et al.* [26], respectively, giving a general tool for a class of mixed channels. We derive a steady-state equation which gives a graphical interpretation of decoder convergence. This allows one to estimate the maximum erasure capability on the mixture channel, or conversely to estimate the additional signal power required to compensate for the loss due to erasures. We see that a good (capacity-approaching) LDPC code over an AWGN channel is also good over the mixed channel up to a moderate erasure probability. We also investigate practical issues such as maximum number of iterations of message-passing decoders, coded block lengths and types of erasure patterns (random/block erasures). We design an optimized LDPC code for the mixed channel, which shows better performance if the erasure probability is larger than a certain value (0.1 in our simulation) at the expense of performance degradation at unerased (AWGN channel) and lower erasure probability regions (less than 0.1 in our simulation)

We also consider puncturing of LDPC codes for additive white Gaussian noise channels. We show that good puncturing patterns exist and that the puncturing can be performed in a rate-compatible fashion. Furthermore, rate-compatible puncturing results in small loss of performance with respect to threshold, namely, the punctured code is good (in terms of threshold) across a range of rates when compared with the

optimal codes for each rate. This allows one to implement a single "mother" encoder and decoder that is good across a wide range of rates.

Finally, we discuss universality of punctured LPDC codes in terms of excess mutual information (EMI) [23, 24]. There is a relation between E_b/N_0 gap and EMI, which tells E_b/N_0 gaps are exaggerated at high data rates for the same EMI, and EMI is a fairer measure to see the universality of LDPC codes. We empirically show that it is possible to design universal LDPC codes (in terms of EMI) across a broad range of coding rates (from 0.1 to 0.95 in our simulations) with the puncturing techniques proposed in this thesis. As a practical issue, we propose a way to design high rate LDPC codes with the puncturing technique, which designs high rate LDPC codes by puncturing low rate LDPC codes. We compare punctured LDPC codes with LDPC codes designed (dedicated LDPC codes) for a high coding rate (0.8 in our simulations). It is observed that the punctured LDPC codes have lower error-floors at high E_b/N_0 regions thanks to longer effective block lengths and sparser party-check matrices. The punctured LDPC codes also have better frame-error rates (FERs) and decoder-error rate (DERs). In addition to lower FERs and DERs, the punctured LDPC codes have an efficient structure for code combining [5, 43] that is a key element of type-II hybrid automatic repeat request (ARQ) protocols together with low FERs and DERs.

CHAPTER I

INTRODUCTION

1.1 Motivation

Since the rediscovery of low-density parity-check (LDPC) codes in the middle 1990's [30], many research results have shown capacity-approaching performances of LDPC codes [27, 26, 38, 37], which culminated with an LDPC code within 0.0045dB of the Shannon limit [7]. Although there has not been any mathematically rigorous work proving LDPC codes eventually achieve the Shannon limit, it is conceivable that LDPC codes really do.

In this thesis, we extend applications of LDPC codes to a combination of constituent sub-channels, which is a mixture of Gaussian channels with erasures [16, 18]. This model, for example, represents a common channel in magnetic recordings where thermal asperities in the system are detected and represented at the decoder as erasures. Although this channel is practically useful, we cannot find any previous work that evaluates performance of LDPC codes over this channel. We are also interested in practical issues such as designing robust LDPC codes for the mixture channel and predicting performance variations due to erasure patterns (random and burst), and finite block lengths.

On time varying channels, a common error control strategy is to adapt the coding rate according to available channel state information (CSI). An effective way to realize this coding strategy is to use a single code and puncture it in a rate-compatible fashion, a so-called rate-compatible punctured code (RCPC) [4, 21]. We are interested in the existence of good puncturing patterns for rate-changes that minimize performance loss. We show the existence of good puncturing patterns with analysis

and verify the results with simulations.

Universality of a channel code across a broad range of coding rates is a theoretically interesting topic. We are interested in the possibility of using the puncturing technique proposed in this thesis for designing universal LDPC codes. We also consider how to design high rate LDPC codes by puncturing low rate LDPC codes. The new design method can take advantage of longer effect block lengths, sparser parity-check matrices, and larger minimum distances of low rate LDPC codes.

1.2 Scope of Thesis

This thesis is organized as follows. In Chapter 2, we introduce basic terminologies of low-density parity-check (LDPC) codes and the sum-product decoding algorithm. We also summarize density evolution and Gaussian approximation that are helpful to understand the sequel sections.

In Chapter 3, we extend Chung's work to include a channel model that is a mixture of Gaussian noise with random erasures. In Section 3.2 we consider regular LDPC codes and define the basic terminology used in the subsequent sections. In Section 3.3 we extend the results of Section 3.2 to irregular LDPC codes. In Section 3.4 we simplify the results of Section 3.3 using the fact that the erasure probability during iterations does not depend on a signal-to-noise ratio. The simplified equation, called a *steady-state equation*, gives us a graphical interpretation of decoder convergence that was originally introduced in [8]. In Section 3.5 we verify our analysis and the channel model by comparing theoretical E_b/N_0 thresholds of regular and irregular LDPC codes with simulation results of actual LDPC codes. We also investigate some practical issues such as maximum number of iterations of message-passing decoders, coded block lengths and types of erasure patterns (random/block erasures). In Section 3.6 we design an LDPC code for the mixed channel and compare the designed LDPC code with an LDPC code optimized for an AWGN channel in [8]. We conclude this

chapter in Section 3.7.

In Chapter 4, we apply a rate-compatible puncturing scheme to low-density parity-check (LDPC) codes, namely one would like to have a single LDPC code which, when punctured in a rate-compatible way, remains good across a range of punctured rates. We present a way to puncture LDPC codes which does not disturb the optimality of the base code, and where the resulting punctured codes maintain threshold optimality across a range of rates. We focus mainly on asymptotic thresholds of the punctured LDPC codes instead of practical issues such as code performance with a short block length, number of iterations to achieve a saturated performance and finite precision effects due to quantization. In Section 4.1, we introduce the channel model we consider and define terminologies for the sequel sections. In Section 4.2, we analyze the thresholds of punctured LDPC codes with Gaussian approximation (GA) in [8]. The analytic results are useful to understand the convergence of the punctured LDPC codes and can be further simplified with a proper assumption. The simplified recursive equation, called a *steady-state equation*, tells us how the punctured LDPC codes perform and how to design the puncturing distributions. Thus, the analysis gives us not only a prediction method of the thresholds of the punctured LDPC codes but also a design rule of optimal puncturing distributions. In Section 4.3, we design puncturing distributions for two LDPC codes which are designed in [37] and [6]. We also implement the LDPC codes with a code block length of 131,072 and apply the designed puncturing distributions to the implemented LDPC codes. Through the simulation, we confirm consistency between the asymptotic and implemented performance. Finally, in Section 4.4, we summarize this chapter.

In Chapter 5, we consider more issues on the proposed punctured LDPC codes such as universality and performance of finite-length punctured LDPC codes for high rates (0.8 in our simulations). In Section 5.1, we investigate universality of punctured LDPC codes over a broad range of coding rates (from 0.1 to 0.95 in our study).

Performance variations are measured in the excess mutual information (EMI) sense instead of E_b/N_0 gap because the EMI is independent of coding rate. Our study shows that universal LDPC codes over a range of coding rates can be implemented with the puncturing technique proposed in Chapter 4. In Section 5.2, we show that high rate LDPC codes can be implemented by puncturing lower rate LDPC codes. We compare performance of the punctured LDPC codes for achieving a coding rate of 0.8 with LDPC codes designed for the same coding rate. Simulations show that punctured LDPC codes have better error-floors, frame-error rate (FER) and decoder-error rate (DER). The lower FER and DER make punctured LDPC codes more amenable to type-II hybrid automatic repeat request (ARQ) protocol. In Section 5.3, we summarize this chapter.

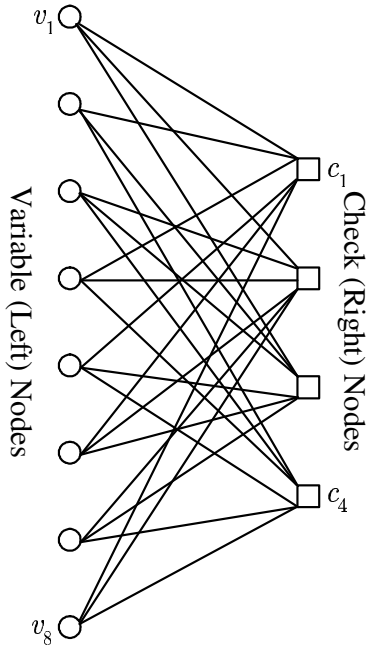
Finally, in Chapter 6, we summarize the work in this thesis and suggest possible future works.

CHAPTER II

BACKGROUND

2.1 Low-Density Parity-Check Codes

Low-Density Parity-Check (LDPC) codes are simple parity check codes [44] defined by parity-check matrices that are restricted to have a small number of 1's in each row and column as compared to the lengths of the column and row. The term, *low-density* implies sparse parity-check matrices. An LDPC code defined by an $r \times n$ sparse parity-check matrix, $\mathbf{H}$ can be also graphically represented by a bipartite graph [41] depicted in Fig. 2.1. On the graph, each column and row of the sparse parity-check matrix are denoted as a *variable* (or *left*) and *check* (or *right*) nodes and represent a received coded symbol and a parity check, respectively. That is, the nth variable (check) node represents the nth column (row) of $\mathbf{H}$.



$$\mathbf{H} = \begin{pmatrix} 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 \end{pmatrix}$$

Figure 2.1: A bipartite graph and parity-check matrix of a (3, 6) regular code.

In a bipartite graph, a connection between a variable node and a check node is called an *edge* [9] which corresponds to a non-zero term at the position indexed by the column and row. Each node has at least one edge, and the number of edges incident with a node is called the *degree* of the node [9]. Thus, if a variable (check) node has d edges, the corresponding column (row) has d non-zero terms in $\mathbf{H}$. A *cycle* is a walk through edges from a node to itself without visiting a node and edge again except the end nodes [9]. In Fig. 2.1, $v_1 - c_1 - v_4 - c_2 - v_1$ makes a cycle of length (*girth*) 4, where v_n (c_n) represents the nth variable (check) node.

While a parity-check matrix and a bipartite graph specify an instance of an LDPC code, an ensemble of LDPC codes can be described with a degree distribution pair $(\lambda(x), \rho(x))$ [27]. $\lambda(x) = \sum_{i=2}^{d_l} \lambda_i x^{i-1}$ ($\rho(x) = \sum_{i=2}^{d_r} \rho_i x^{i-1}$) is a polynomial whose coefficients are non-negative real numbers, the sum of the coefficients is equal to 1, and λ_i (ρ_i) is the fraction of edges belonging to variable (check) nodes with i edges (or a *degree* of i), and d_l (d_r) is the maximum variable (check) degree. Thus, we can have many different parity-check matrices which comply with a degree distribution pair, and either a parity-check matrix or a bipartite graph is a realization of the degree distribution pair.

The coding rate of LDPC codes specified by a degree distribution pair, $(\lambda(x), \rho(x))$ is computed as (see [37])

$$r(\lambda, \rho) = 1 - \frac{\int_0^1 \rho(x)dx}{\int_0^1 \lambda(x)dx} = 1 - \frac{\sum_{j=2}^{d_r} \rho_j / j}{\sum_{j=2}^{d_l} \lambda_j / j}.$$

The degree distribution from an edge perspective can be converted to a node perspective with the following relations:

$$\lambda'_j = \frac{\lambda_j/j}{\sum_{i=2}^{d_l} \lambda_i/i} \Leftrightarrow \lambda_j = \frac{j\lambda'_j}{\sum_{i=2}^{d_l} i\lambda'_i} \text{ and } \rho'_j = \frac{\rho_j/j}{\sum_{i=2}^{d_r} \rho_i/i} \Leftrightarrow \rho_j = \frac{j\rho'_j}{\sum_{i=2}^{d_r} i\rho'_i},$$

where λ'_j (ρ'_j) is the fraction of variable (check) nodes having j edges.

For a given degree distribution pair ($\lambda(x)$ and $\rho(x)$), the specific locations of non-zero terms in a parity-check matrix and degrees of columns and rows are randomly generated in compliance with $\lambda(x)$ and $\rho(x)$, respectively. However, the random generation does not necessarily guarantee a good LDPC code, and exhaustive search in some cases may be prohibitive due to the size of an ensemble. Unfortunately, there does not exist a general and systematic approach to design a good LDPC code out of an ensemble. There have been several attempts to address the design issue in [42, 10].

We will introduce how to evaluate the averaged behavior and performance of LDPC codes in an ensemble during message-passing decoding in the next section.

2.2 Decoding Algorithms

Decoding is a decision process which finds a codeword that minimizes the probability of decoder error based on a received word $\mathbf{r}$ [44]. The minimization is equivalent to choosing a codeword that maximizes a posteriori probability (MAP) which is called MAP decoding. That is, the MAP decoder finds the codeword $\mathbf{c}_j$ such that

$$\max_{\mathbf{c}_j \in \mathbf{C}} \Pr\{\mathbf{c}_j|\mathbf{r}\},$$

where $\mathbf{c}_j$ is a code in an entire code set $\mathbf{C}$. However, the size of the code set, $|\mathbf{C}|$ grows exponentially with a block length which makes an exhaust search practically impossible. Thus, we need a more systematic way to find the most likelihood codeword.

Gallager [14, 15] proposed several iterative decoding algorithms for Low-Density

Parity-Check (LDPC) codes on Binary Erasure Channel (BEC). The proposed algorithms are *message-passing algorithms* which iteratively pass messages between nodes through edges in a bipartite graph. The message is reliability information which can be a probability for a node to be a symbol. For example, if a code is defined over GF(2), the message can be a probability for a node to be '0' (or equivalently '1').

MacKay and Neal [30, 29] rediscovered and decoded LDPC codes on Additive White Gaussian Noise (AWGN) channel with the *belief-propagation* (BP) algorithm (also known as the *sum-product algorithm*) [34] which also works with the framework of the *message-passing algorithm*. The BP algorithm makes exactly the same result as the MAP decoder when a bipartite graph contains no short cycles [34] and received symbols are independent of each other.

The BP algorithm is equivalently described in probability and log-probability domains, although the latter may be more favorable for hardware implementation. We summarize the algorithm based on the [30, 29, 13]:

Belief Propagation Algorithm in Probability Domain

1. **Definitions:**

   - $\mathbf{c} = (c_1, c_2, \ldots c_N)$ is a codeword of a length N, the occurrence of each codeword out of the entire code set ($\mathbf{C}$) is equally probable, $c_n \in$ GF(2) for $1 \leq n \leq N$, $\mathbf{r} = (r_1, r_2, \ldots, r_N)$ is a received symbol vector whose elements, r_n's are independent each other, $\Pr(\mathbf{r}|\mathbf{c}) = \prod_{n=1}^{N} \Pr(r_n|c_n)$, and r_n corresponds to the nth variable node in a bipartite graph defining an LDPC code.
   - $f_n^x = \Pr(r_n|c_n = x)$, and $f_n^1 = 1 - f_n^0$.
   - $\mathcal{M}(n) = \{m : h_{mn} = 1\}$ ($\mathcal{N}(m) = \{n : h_{mn} = 1\}$) is a set of check (variable) node indices adjacent to the variable node n (the check node m),

and h_{mn} is the element located at the mth row and the nth column of the parity check matrix $\mathbf{H}$.

- $\mathcal{M}(n) \setminus m$ ($\mathcal{N}(m) \setminus n$) is the set $\mathcal{M}(n)$ ($\mathcal{N}(m)$) less the index m (n).
- $q_{mn}^{x,(\ell)}$ is the probability for c_n to be x at the ℓth iteration, which is the message from the nth variable to the mth check nodes.
- $r_{mn}^{x,(\ell)}$ is the probability that the check m is satisfied at the ℓth iteration if c_n is x and those of the other variable nodes involved in the decision of $r_{mn}^{x,(\ell)}$ are given by $\{q_{mn'}^{x,(\ell-1)} : n' \in \mathcal{N}(m) \setminus n, \text{ and } x \in \mathrm{GF}(2)\}$, which is the message from the mth check to the nth variable nodes.

2. **Initialization:**

   For each m and n satisfying $h_{mn} = 1$, which is valid through all the steps, initialize $q_{mn}^{x,(0)} = f_n^x$, for $x \in \mathrm{GF}(2)$.

3. **Check node update:**

   Compute

$$\begin{aligned} \delta r_{mn}^{(\ell)} &= \prod_{n' \in \mathcal{N}(m) \setminus n} \delta q_{mn'}^{(\ell-1)} \\ r_{mn}^{x,(\ell)} &= \frac{1}{2}\left(1 + (-1)^x \delta r_{mn}^{(\ell)}\right) \end{aligned}$$

   where $\delta q_{mn}^{(\ell)} = q_{mn}^{0,(\ell)} - q_{mn}^{1,(\ell)}$ and $\ell \in \mathbb{N}$.

4. **Variable node updates:**

   For each variable node index n from 1 to N

$$\begin{aligned} q_{mn}^{x,(\ell)} &= \alpha_{mn}^{(\ell)} f_n^x \prod_{m' \in \mathcal{M}(n) \setminus m} r_{m'n}^{x,(\ell)}, \\ q_n^{x,(\ell)} &= \alpha_n^{(\ell)} f_n^x \prod_{m' \in \mathcal{M}(n)} r_{m'n}^{x,(\ell)}, \end{aligned}$$

where $\alpha_{mn}^{(\ell)}$ and $\alpha_{n}^{(\ell)}$ are chosen to make $q_{mn}^{0,(\ell)} + q_{mn}^{1,(\ell)} = 1$ and $q_{n}^{0,(\ell)} + q_{n}^{1,(\ell)} = 1$, respectively.

5. **Verify parity checks:**

   For each n, if $q_n^{0,(\ell)} > 0.5$ then $\hat{c}_n = 0$ otherwise $\hat{c}_n = 1$. If $\mathbf{H}^T\hat{\mathbf{c}} = \mathbf{0}$, then $\hat{\mathbf{c}}$ becomes the decoder output and halt the algorithm, otherwise go to 3, where $\hat{c}_n$ is the nth decoded bit and $\hat{\mathbf{c}}$ is the estimated codeword. Practically, the loop from 3 to 5 repeats a preset number of times. If we cannot find a valid codeword within the number of iterations, we have to declare a *decoder failure* [44].

Belief Propagation Algorithm in Log-Probability Domain

1. **Definitions:**

   - $f_n = \ln\left(\Pr(c_n = 0|\mathbf{r})/\Pr(c_n = 1|\mathbf{r})\right)$
   - For any $x \in (-\infty, +\infty)$, a map $\gamma : [-\infty, +\infty] \to \mathrm{GF}(2) \times [0, +\infty]$ is defined as

     $$\gamma(x) := (\gamma_1(x), \gamma_2(x)) := (\mathrm{sgn}(x), \Psi(x)),$$

     where

     $$\Psi(x) := \begin{cases} -\ln\tanh\left|\frac{x}{2}\right|, & \text{for } x \in [-\infty, 0) \cup (0, +\infty] \\ +\infty, & \text{for } x = 0 \end{cases}$$

     $$\mathrm{sgn}(x) := \begin{cases} 0, & \text{if } x > 0 \\ 0, & \text{with probability } \frac{1}{2} \text{ if } x = 0 \\ 1, & \text{with probability } \frac{1}{2} \text{ if } x = 0 \\ 1, & \text{if } x < 0. \end{cases}$$

   - $\gamma(z) = \gamma(x) + \gamma(y)$ is defined as $\gamma_1(z) = \gamma_1(x) + \gamma_1(y)$ and $\gamma_2(z) = \gamma_2(x) + \gamma_2(y)$ over $\mathrm{GF}(2)$ and $[0, +\infty]$, respectively.

- For $\gamma(x) = (\chi_1, \chi_2) \in \text{ GF}(2) \times [0, \infty]$,

$$x = \gamma^{-1}(\gamma(x)) := \begin{cases} \Psi(\chi_2), & \text{for } \chi_1 = 0, \\ -\Psi(\chi_2), & \text{for } \chi_1 = 1, \end{cases}$$

because $\Psi(\Psi(x)) = |x|$ and $\chi_1 = 1$ for $x < 0$.

2. **Initialization:**

   For each m and n satisfying $h_{mn} = 1$, define $q_{mn}^{(0)} = f_n$.

3. **Check node updates:**

$$r_{mn}^{(\ell)} = \gamma^{-1}\left(\sum_{n' \in \mathcal{N}(m) \backslash n} \gamma(q_{mn'}^{(\ell-1)})\right), \tag{2.1}$$

   where $\ell \in \mathbb{N}$.

4. **Variable node updates:**

$$\begin{aligned} q_n^{(\ell)} &= f_n + \sum_{m \in \mathcal{M}(n)} r_{mn}^{(\ell)}, \text{ and} \\ q_{mn}^{(\ell)} &= q_n^{(\ell)} - r_{mn}^{(\ell)} \\ &= f_n + \sum_{m' \in \mathcal{M}(n) \backslash m} r_{m'n}^{(\ell)}. \end{aligned} \tag{2.2}$$

5. **Verify parity checks:**

   For each n, if $q_n^{(\ell)} > 0$ then $\hat{c}_n = 0$, otherwise $\hat{c}_n = 1$. If $\mathbf{H}^T \hat{\mathbf{c}} = \mathbf{0}$, then $\hat{\mathbf{c}}$ becomes the decoder output and halt the algorithm, otherwise go to 3, where $\hat{c}_n$ is the nth decoded bit and $\hat{\mathbf{c}}$ is the estimated codeword. Practically, the loop from 3 to 5 repeats a preset number of times. If we cannot find a valid codeword within the number of iterations, we have to declare a *decoder failure* [44].

In principle, the BP algorithm is optimal but in some situations, the algorithm requires two stringent assumptions: 1) no cycle in the bipartite graph, 2) exact knowledge of the soft information (probability distribution function/log-likelihood ratio). There have been several attempts to address these practical issues in [12, 13, 32].

2.3 Density Evolution

Richardson *et al.* [38, 37] computed the asymptotic (block length tends to infinity) behavior of Belief-Propagation (BP) decoders averaged over all Low-Density Parity-Check (LDPC) codes in an ensemble specified by a degree distribution pair. They traced the variations of the probability density functions (*density* for short) of messages between variable and check nodes in the BP decoder during iterations, which is called *density evolution*. Although density evolution extensively counts on numerical computations, it is a useful tool not only to evaluate *thresholds* (will be defined shortly) of LDPC codes but also to design good LDPC codes for many different channels.

Density evolution requires the following conditions;

1. symmetry condition:

   the channel output must be symmetric, and

   [Definition 1 in [37]], the density f is symmetric if

   $$f(x) = e^x f(-x), \tag{2.3}$$

   for $x \in \mathbb{R}$,

2. No short cycles:

   the bipartite graph contains no short cycles.

The first requirement with the symmetry of the BP algorithm [38] frees the behavior of the BP decoding from the channel input patterns (see Lemma 1 in [38]). Thus, behavior of the BP algorithm with a simple input pattern such as all zeros expressed with positive ones (equivalently, all ones expressed with negative ones) is exactly the same as that of any other input patterns. In the explanation of density evolution, the all-zero code is assumed. The second requirement is a necessary condition for the BP decoding results to be those of the MAP decoding [34]. The requirement can

be met by increasing a block length to infinity [37], which is impossible in practical situations. It seems to be pessimistic because density evolution only tells behaviors of LDPC codes with infinite block lengths. However, they came to very useful and important conclusions [38];

1. [Concentration] *"Let $P_e^n(\ell)$ be the expected fraction of incorrect messages which are passed in the ℓth iteration, where the expectation is over all instances of the code, the choice of the message, and the realization of the noise. For any $\delta > 0$, the probability that the actual fraction of incorrect messages which are passed in the ℓth iteration for any particular such instance lies outside the range $(P_e^n(\ell) - \delta, P_e^n(\ell) + \delta)$ converges to zero exponentially fast in n."*

   In other words, if we pick up an LDPC code from an ensemble defined by a degree distribution pair, the performance of the chosen code concentrates around an average value. As a block length, n increases, the performance converges to the average value exponentially fast in n.

2. [Convergence to Cycle-Free Case] *"$P_e^n(\ell)$ converges to $P_e^\infty(\ell)$ as n tends to infinity, where $P_e^\infty(\ell)$ is the expected fraction of incorrect messages passed in the ℓth decoding round assuming that the graph does not contain cycles of length 2ℓ or less."*

   This conclusion does not tell us how fast LDPC codes converge to the performance predicted by density evolution, $P_e^\infty(\ell)$. However, they have seen empirically the convergence is fast enough in some applications.

3. [Density Evolution and Threshold Determination] *"$P_e^\infty(\ell)$ is computable by a deterministic algorithm. Furthermore, there exists a channel parameter, σ^*, the threshold with the following property: if $\sigma < \sigma^*$ then $\lim_{\ell\to\infty} P_e^\infty(\ell) = 0$; if, on the other hand, $\sigma > \sigma^*$, then there exists a constant $\gamma(\sigma) > 0$ such that $P_e^\infty(\ell) > \gamma(\sigma)$ for all $\ell \geq 1$."*

We can apply optimization techniques [36, 45] for a degree distribution pair to find the largest threshold for a given coding rate, which is a design process of good LDPC codes.

We introduce the original density evolution [38, 37] and discretized density evolution [7, 6] which is faster and closer to practical BP decoders.

Density Evolution

1. **Definitions:**

   The symbols P_ℓ and Q_ℓ denote the densities of the random variables $m_{\mathrm{vc}}^{(\ell)}$ (the message from variable to check nodes) and $m_{\mathrm{cv}}^{(\ell)}$ (the message from check to variable nodes), respectively. P_0 is the density of the channel output, m_0 (the message from a channel).

2. **Check node update:**

   Recall the check node updating rule of the BP decoding algorithm in the log-probability domain;

   $$r_{mn}^{(\ell)} = \gamma^{-1}\left(\sum_{n' \in \mathcal{N}(m)\backslash n} \gamma\left(q_{mn'}^{(\ell-1)}\right)\right),$$

   which is the message from a check to a variable node given the check node has a degree of $|\mathcal{N}(m)|$. The message averaged over a given degree distribution can be expressed as

   $$m_{\mathrm{cv}}^{(\ell)} = \sum_{i \geq 2} \rho_i \gamma^{-1}\left(\sum_{j=1}^{i-1} \gamma\left(m_{\mathrm{vc},j}^{(\ell-1)}\right)\right),$$

   where $\gamma\left(m_{\mathrm{vc},j}^{(\ell-1)}\right)$'s are independent and identically distributed (i.i.d.) random variables ($\equiv \gamma\left(m_{\mathrm{vc}}^{(\ell-1)}\right)$) and the density of the summation of the random variables becomes the convolution of densities of the random variables over $\mathrm{GF}(2) \times [0, +\infty]$. To define a convolution of functions over $\mathrm{GF}(2) \times [0, +\infty]$, γ

is equivalently expressed in a different form;

$$\gamma\left(s, \Psi(x)\right) \equiv \chi_{\{s=0\}}\gamma^0 + \chi_{\{s=1\}}\gamma^1,$$

where for $x \in (-\infty, +\infty]$, $s = \mathrm{sgn}(x)$, $\gamma^s = \Psi\left((-1)^s x\right)$ is a function over $[0, +\infty]$ and $\chi_{\{s=a\}} = 1$ if $s = a$ and $\chi_{\{s=a\}} = 0$, otherwise. For two functions g and h over $\mathrm{GF}(2) \times [0, \infty]$, the convolution is defined as

$$\begin{aligned} g \otimes h &= \left(\chi_{\{s=0\}} g^0 + \chi_{\{s=1\}} g^1\right) \otimes \left(\chi_{\{s=0\}} h^0 + \chi_{\{s=1\}} h^1\right) \\ &:= \chi_{\{s=0\}}\left(g^0 \otimes h^0 + g^1 \otimes h^1\right) + \chi_{\{s=1\}}\left(g^1 \otimes h^0 + g^0 \otimes h^1\right). \end{aligned}$$

The density of $m_{\mathrm{cv}}^{(\ell)}$ is computed as

$$Q_\ell = \Gamma^{-1}\left(\rho\left(\Gamma\left(P_{\ell-1}\right)\right)\right) := \Gamma^{-1}\left(\sum_{i\geq 2} \rho_i \left(\Gamma\left(P_{\ell-1}\right)\right)^{\otimes(i-1)}\right), \tag{2.4}$$

where n time convolutions of the density $(P_{\ell-1})$ are shortened as $P_{\ell-1}^{\otimes(n)}$, Γ transforms the density of a random variable x to the corresponding density of $\gamma(x)$ and Γ^{-1} vice versa (see the details in [37]).

3. **Variable node update:**

Recall the variable node updating rule of the BP decoding algorithm in the log-probability domain;

$$q_{mn}^{(\ell)} = \begin{cases} f_n, & \text{if } \ell = 0 \\ f_n + \displaystyle\sum_{m' \in \mathcal{M}(n)\backslash m} r_{m'n}^{(\ell)} & \text{if } \ell \geq 1, \end{cases}$$

where $q_{mn}^{(\ell)}$ represents the message from variable to check nodes which is decided by the channel message, f_n, and the messages from the check nodes. Each variable node has different numbers of check nodes, and the average of the message over a degree distribution can be computed as

$$m_{\mathrm{vc}}^{(\ell)} = \begin{cases} m_0, & \text{if } \ell = 0 \\ m_0 + \displaystyle\sum_{i\geq 2} \lambda_i \sum_{j=1}^{i-1} m_{\mathrm{cv},j}^{(\ell)} & \text{if } \ell \geq 1, \end{cases}$$

and the density is represented as

$$P_\ell = P_0 \otimes \lambda(Q_\ell) := P_0 \otimes \sum_{i \geq 2} \lambda_i (Q_\ell)^{\otimes(i-1)}, \tag{2.5}$$

where m_0 is the log-likelihood ratio of the channel output, the density of the sum of i.i.d. random variables ($m_{\mathrm{cv},j}^{(\ell)}$'s $\equiv m_{\mathrm{cv}}^{(\ell)}$), is expressed as the convolution of the density (Q_ℓ) of each random variable [3], $P_0 \otimes Q_\ell$ is the convolution between the two densities (P_0 and Q_ℓ), and $Q_\ell^{\otimes(i-1)}$ represents the $(i-1)$ convolutions of Q_ℓ.

4. **Message error fraction:**

   Due to symmetry, the behavior of the BP decoding does not depend on channel input patterns. In the density evolution, all positive ones (all zero codeword) are assumed to be transmitted (all negative ones are also possible). Thus, the fraction of message error is the proportion of the variable to check node message that is less than zero;

$$P_e^\infty(\ell) = \lim_{\delta \to 0} \int_{-\infty}^{-\delta} P_\ell(m) dm + \frac{1}{2} P_\ell(0), \text{ for } \delta > 0.$$

   If a channel parameter (σ) is less than a threshold (σ^*), the error fraction ($P_e^\infty(\ell)$) monotonically decreases toward zero.

By tracing the variations of the densities (P_ℓ in (2.5) and Q_ℓ in (2.4)) during iterations, we can analyze the convergence of BP decoders for LDPC codes and evaluate the thresholds. To iteratively compute the densities with digital computers, we have to be careful in quantizing the densities during iterations due to the nonlinearity of the transformations, Γ and Γ^{-1}. Small quantization error can be accumulated and lead to inaccurate results. Another difficulty to be overcome is the complexity involved in the convolutions, which has a square growth with the bit precision of the quantization. Although density evolution describes theoretical behaviors of LDPC codes

with BP decoders, the results are different from those out of practical BP decoders that have a finite internal precision, a simple quantization before/after the nonlinear transformations and an upper and lower limits of the magnitudes of the messages.

Chung [6, 7] introduced a modified version of density evolution called *Discretized Density Evolution* (DDE) which models more accurately practical BP decoders and speeds up the computation by taking advantage of computational symmetry and nesting-based computations. Furthermore, the nonlinear transformations in density evolution can be realized with a mapping table. We introduce DDE based on [6, 7].

Discretized Density Evolution

1. **Quantization:**

   Quantize the message with the center-tread uniform quantizer;

$$\mathcal{Q}(m) = \begin{cases} \lfloor \frac{m}{\Delta} + \frac{1}{2} \rfloor \cdot \Delta, & \text{if } w \geq \frac{\Delta}{2} \\ \lceil \frac{m}{\Delta} - \frac{1}{2} \rceil \cdot \Delta, & \text{if } w \leq -\frac{\Delta}{2} \\ 0, & \text{otherwise ,} \end{cases}$$

   which has a symmetric output, i.e., $\mathcal{Q}(m) = -\mathcal{Q}(m)$, Δ is the quantization step $\lfloor x \rfloor$ ($\lceil x \rceil$) is the largest (smallest) integer not greater (less) than x. The largest and smallest message values are $\left(2^{(Q_b-1)} - 1\right) \cdot \Delta$ and $-2^{Q_b-1} \cdot \Delta$ given Q_b bit precision. The quantized message is still a random variable and is described by a probability mass function, $p_{\bar{m}}[k]$ that is related to the probability density functions as

$$p_{\bar{m}}[k] = \Pr[\bar{m} = k\Delta] = \int_{k\Delta - \frac{\Delta}{2}}^{k\Delta + \frac{\Delta}{2}} f_m(\mu) d\mu,$$

   where $f_m(\mu)$ is the density of m, $\bar{m}$ is the quantized message and $p_{\bar{m}}[k]$ is the probability mass function (pmf) of $\bar{m}$.

2. **Check node update:**

   In density evolution, computing $m_{\text{cv}}^{(\ell)}$ is the most time consuming part due to

the nonlinear transformations, (γ and γ^{-1}). This computational difficulty can be alleviated by using the following two functions;

- $\mathcal{R}_2 : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ is defined as $\mathcal{R}_2(a,b) := \gamma^{-1}\left(\gamma(a,b)\right)$,
- $\mathcal{R} : \mathbb{R} \times \mathbb{R} \to \mathbb{Q}_{Q_b}$ is defined as $\mathcal{R}(a,b) := \mathcal{Q}\left(\mathcal{R}_2(a,b)\right)$,
  where $\mathbb{Q}_{Q_b}$ is $\left\{j \cdot \Delta : -2^{(Q_b-1)} \le j \le 2^{(Q_b-1)} - 1 \text{ and } j \in \mathbb{Z}\right\}$.

$\gamma^{-1}\left(\sum_{j=1}^{n} \gamma(a_j)\right)$ can be easily represented as $\mathcal{R}_2\left(a_1, \mathcal{R}_2\left(a_2, \ldots, \mathcal{R}_2\left(a_{n-1}, a_n\right)\cdots\right)\right)$ and the message,

$$m_{\mathrm{cv}}^{(\ell)} = \sum_{i\ge 2} \rho_i \gamma^{-1}\left(\sum_{j=1}^{i-1} \gamma\left(m_{\mathrm{vc},j}^{(\ell-1)}\right)\right),$$

can be rewritten as

$$m_{\mathrm{cv}}^{(\ell)} = \sum_{i\ge 2} \rho_i \mathcal{R}_2\left(m_{\mathrm{vc},1}^{(\ell-1)}, \mathcal{R}_2\left(\ldots, \mathcal{R}_2\left(m_{\mathrm{vc},i-2}^{(\ell-1)}, m_{\mathrm{vc},i-1}^{(\ell-1)}\right)\cdots\right)\right).$$

The quantized message, $\bar{m}_{\mathrm{cv}}^{(\ell)}$ is

$$\begin{aligned}
\bar{m}_{\mathrm{cv}}^{(\ell)} &= \sum_{i\ge 2} \rho_i \mathcal{R}\left(\bar{m}_{\mathrm{vc},1}^{(\ell-1)}, \mathcal{R}\left(\ldots, \mathcal{R}\left(\bar{m}_{\mathrm{vc},i-2}^{(\ell-1)}, \bar{m}_{\mathrm{vc},i-1}^{(\ell-1)}\right)\cdots\right)\right) \\
&\equiv \sum_{i\ge 2} \rho_i \underbrace{\mathcal{R}\left(\bar{m}_{\mathrm{vc}}^{(\ell-1)}, \mathcal{R}\left(\ldots, \mathcal{R}\left(\bar{m}_{\mathrm{vc}}^{(\ell-1)}, \bar{m}_{\mathrm{vc}}^{(\ell-1)}\right)\cdots\right)\right)}_{i-1 \text{ time nesting}} \\
&= \sum_{i\ge 2} \rho_i \mathcal{R}^{i-1} \bar{m}_{\mathrm{vc}}^{(\ell-1)},
\end{aligned}$$

where $\mathcal{R}\left(\bar{m}_{\mathrm{vc}}^{(\ell-1)}, \mathcal{R}\left(\ldots, \mathcal{R}\left(\bar{m}_{\mathrm{vc}}^{(\ell-1)}, \bar{m}_{\mathrm{vc}}^{(\ell-1)}\right)\cdots\right)\right)$ is shortened as $\mathcal{R}^{i-1}\bar{m}_{\mathrm{vc}}^{(\ell-1)}$. $\mathcal{R}(\bar{a}, \bar{b})$ is only defined at the $2^{Q_b} \times 2^{Q_b} = 2^{2Q_b}$ points which are implemented with a mapping table whose size is the same as that of the domain, $2^{2Q_b} = 2^{Q_b} \times 2^{Q_b}$. Each content of the table is represented with Q_b bits that are the size of the range, $\mathcal{Q}_{Q_b}$. However, by taking advantage of the symmetries of $\mathcal{R}$; 1) $\mathcal{R}(a,b) = \mathcal{R}(b,a)$, 2) $\mathcal{R}(a,b) = \mathcal{R}(-a,-b)$, 3) $-\mathcal{R}(a,b) = \mathcal{R}(a,-b) = \mathcal{R}(-a,b)$, we can reduce the size of the table by an eighth as shown in Fig. 2.2.

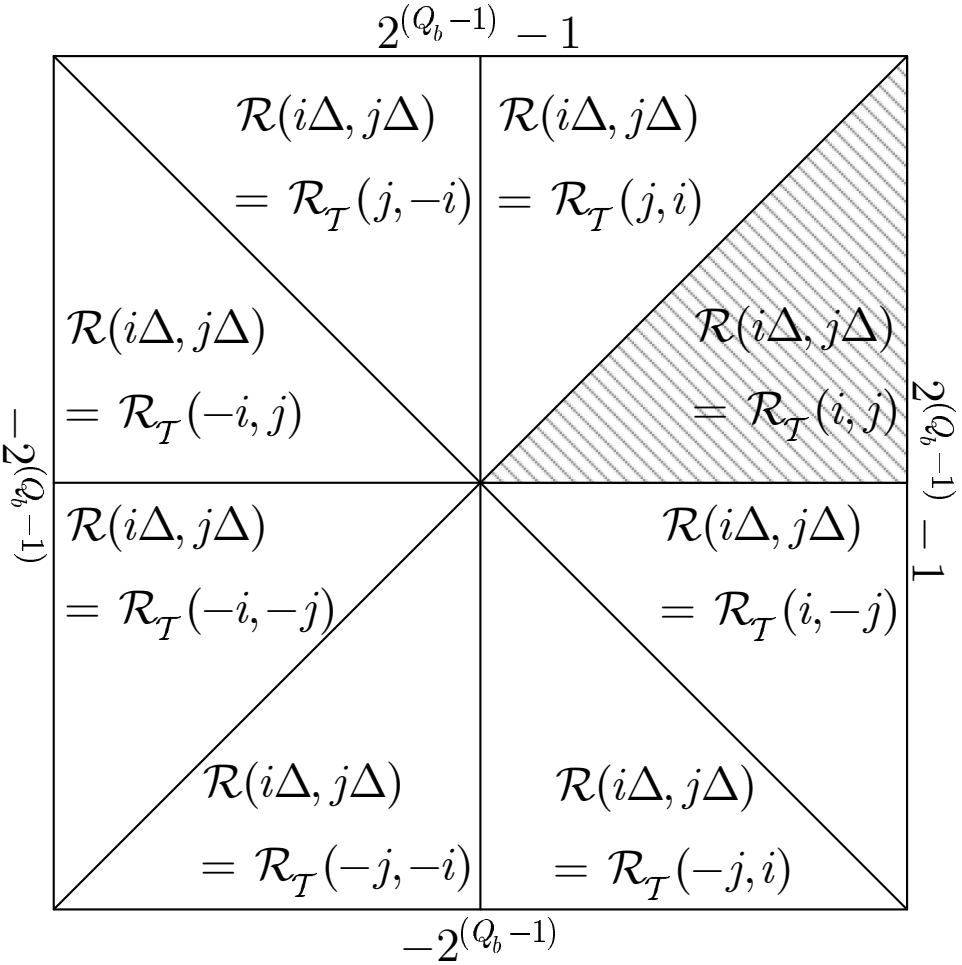


Figure 2.2: Symmetries of $\mathcal{R}(i\Delta, j\Delta)$; the shadowed area will be computed and stored in the table, $\mathcal{R}_T(i,j)$ and the other areas are computed with $\mathcal{R}_T(i,j)$, for $i,j \in \{-2^{Q_b-1}, -2^{Q_b-1}+1, \ldots, 2^{Q_b-1}-1\}$.

The pmf of the message from a check node of degree 3 to a variable node can be computed as

$$q^{(\ell)}_{\bar{m}_{\mathrm{cv}}|d_r=3}[k] = \sum_{(i,j):k\Delta=\mathcal{R}(i\Delta,j\Delta)} p^{(\ell-1)}_{\bar{m}_{\mathrm{vc}}}[i]p^{(\ell-1)}_{\bar{m}_{\mathrm{vc}}}[j],$$

which is the sum of the probabilities for $\mathcal{R}(i\Delta, j\Delta)$ to be $k\Delta$ given the pmf of the message from a variable node at the $(\ell-1)$th iteration, $p^{(\ell-1)}_{\bar{m}_{\mathrm{vc}}}[j]$. For an arbitrary check node of degree d_c, the simplest way to compute the density is

$$q^{(\ell)}_{\bar{m}_{\mathrm{cv}}|d_r=d_c}[k] = \sum_{(j_1,j_2,\ldots,j_{d_c-1})\in\mathbb{A}(k)} p^{(\ell-1)}_{\bar{m}_{\mathrm{vc}}}[j_1]p^{(\ell-1)}_{\bar{m}_{\mathrm{vc}}}[j_2]\cdots p^{(\ell-1)}_{\bar{m}_{\mathrm{vc}}}[j_{d_c-1}],$$

where $\mathbb{A}(k) = \{(j_1, j_2, \ldots, j_{d_c-1}) : k\Delta = \mathcal{R}(j_1\Delta, \mathcal{R}(\ldots, \mathcal{R}(j_{d_c-2}\Delta, j_{d_c-1}\Delta)\cdots))\}$. However, the pmf can also be computed as

$$q^{(\ell)}_{\bar{m}_{\mathrm{cv}}|d_r=d_c}[k] = \sum_{(i,j):k\Delta=\mathcal{R}(i\Delta,j\Delta)} q^{(\ell)}_{\bar{m}_{\mathrm{cv}}|d_r=d_{r_1}}[i]q^{(\ell)}_{\bar{m}_{\mathrm{cv}}|d_r=d_{r_2}}[j], \tag{2.6}$$

where $d_{r_1} + d_{r_2} = d_c + 1$.

A careful scheduling of the *nesting-based computation* makes the algorithm more efficient. For example, if we have the densities of the messages from a degree 3 and a 9 check nodes, we can make the density of message from a degree 11 check node with the given densities. The pmf of $\bar{m}_{\mathrm{cv}}$ is expressed as the weighted sum of pmf's which is

$$\begin{aligned} q^{(\ell)}_{\bar{m}_{\mathrm{cv}}} &= \sum_{i\geq 2}\left\{P[d_r = i]q^{(\ell)}_{\bar{m}_{\mathrm{cv}}|d_r=i}\right\} \\ &= \sum_{i\geq 2}\rho_i q^{(\ell)}_{\bar{m}_{\mathrm{cv}}|d_l=i} \end{aligned}.$$

3. **Variable node update:**

   The pmf of a variable node message can be computed simply with convolutions as

$$p^{(\ell)}_{\bar{m}_{\mathrm{vc}}} = p_{\bar{m}_0} \otimes \lambda(q^{(\ell)}_{\bar{m}_{\mathrm{cv}}}) = p_{\bar{m}_0} \otimes \sum_{i\geq 2}\lambda_i (q^{(\ell)}_{\bar{m}_{\mathrm{cv}}})^{\otimes(i-1)},$$

where $p_{\bar{m}_0}$, $p^{(\ell)}_{\bar{m}_{\mathrm{vc}}}$, and $q^{(\ell)}_{\bar{m}_{\mathrm{cv}}}$ are pmf's of $\bar{m}_0$, $\bar{m}^{(\ell)}_{\mathrm{vc}}$ and $\bar{m}^{(\ell)}_{\mathrm{cv}}$, respectively. The convolution can be efficiently computed using the Fast-Fourier Transformation (FFT). In computing the convolutions, a careful scheduling of the nest-based computation reduces computation time. For example, 9 time convolutions, $\left(q^{(\ell)}_{\bar{m}_{\mathrm{cv}}}\right)^{\otimes 9}$ can be nested as

$$\left(q^{(\ell)}_{\bar{m}_{\mathrm{cv}}}\right)^{\otimes 9} = q^{(\ell)}_{\bar{m}_{\mathrm{cv}}} \otimes \left(\left(\left(q^{(\ell)}_{\bar{m}_{\mathrm{cv}}}\right)^{\otimes 2}\right)^{\otimes 2}\right)^{\otimes 2}.$$

2.4 Gaussian Approximation

Our work is based on the results of *Gaussian Approximation* (GA) in [8]. In this section we briefly summarize GA. This summary is the minimum needed to understand our work and describes GA for irregular LDPC codes only, which includes regular LDPC codes as a special case.

The LLR (log-likelihood ratio) message (*message* for short) of a check node at the kth iteration ($u^{(k)}$ and $k \in \mathbb{N}$) during message-passing decoding is approximated as a Gaussian probability density function that is completely specified by its mean ($m_u^{(k)}$) and variance ($\mathrm{Var}(u^{(k)})$). The symmetry condition, $f(x) = e^x f(-x)$, which is preserved during iterations, imposes a relation between the mean and variance, $\mathrm{Var}(u^{(k)}) = 2m_u^{(k)}$. Thus, by tracing the changes of the mean (called *updated mean*) during iterations, we can watch the evolution of the density of the check node message, $f^{(k)}(u) = \mathcal{N}(m_u^{(k)}, 2m_u^{(k)})$, where $\mathcal{N}(\mu, \sigma^2)$ is the normal probability density function with a mean and variance of μ and σ^2, respectively.

The probability density function (*density* for short) of a variable node message at the kth iteration ($v^{(k)}$) is approximated as a Gaussian mixture for irregular LDPC codes and as Gaussian for regular LDPC codes:

$$f^{(k)}(v) = \sum_{i=2}^{d_l} \lambda_i \mathcal{N}\left(m_{v,i}^{(k)}, 2m_{v,i}^{(k)}\right),$$

where λ_i is the fraction of edges belonging to degree i variable nodes, $m_{v,i}^{(k)} = m_{u_0} + (i-1)m_u^{(k-1)}$, m_{u_0} is the mean of an LLR message out of channel and $m_u^{(0)} = 0$.

Between a check node of degree d_c and the variable nodes, there is the well-known tanh rule [26]:

$$\tanh \frac{u^{(k)}}{2} = \prod_{j=1}^{d_c-1} \tanh \frac{v_j^{(k)}}{2}, \tag{2.7}$$

where $v_j^{(k)}$, $j = 1, \ldots, d_c - 1$, are the incoming messages from $d_c - 1$ neighbors and $u^{(k)}$ is the message sent to the remaining node. The expectation of each side of (2.7) can be simplified as

$$E\left[\tanh \frac{u^{(k)}}{2}\right] = E\left[\tanh \frac{v^{(k)}}{2}\right]^{d_c-1}. \tag{2.8}$$

Because the variable node messages are assumed to be i.i.d., we can omit the index j in (2.7) and simplify the product. The expectation $E\left[\tanh \frac{u}{2}\right]$ can be expressed as

$$E\left[\tanh \frac{u}{2}\right] = \frac{1}{\sqrt{4\pi m_u}} \int_{\mathbb{R}} \tanh \frac{u}{2} e^{\frac{-(u-m_u)^2}{4m_u}} du, \tag{2.9}$$

where u is a Gaussian, $m_u = E[u]$ and $\mathrm{Var}(u) = 2m_u$. Thus, both sides of (2.8) become integral forms that are simplified by the following definition.

Definition 1 (*definition 1 in [8]*:)

$$\phi(x) = \begin{cases} 1 - \dfrac{1}{\sqrt{4\pi x}} \displaystyle\int_{\mathbb{R}} \tanh \frac{u}{2} e^{\frac{-(u-x)^2}{4x}} du, & \text{if } x > 0 \\ 1, & \text{if } x = 0. \end{cases}$$

$\phi(x)$ is continuous and monotonically decreasing on $[0, \infty)$, with $\phi(0) = 1$ and $\phi(\infty) = 0$.

By applying (2.9) and $\phi(x)$ to both sides of (2.8), we can calculate the updated mean of a degree j check node as

$$m_{u,j}^{(k)} = \phi^{-1}\left(1 - \left[1 - \sum_{i=2}^{d_l} \lambda_i \phi\left(m_{v,i}^{(k)}\right)\right]^{(j-1)}\right).$$

The check node updated mean becomes a recursive equation,

$$\begin{aligned} m_u^{(k)} &= \sum_{j=2}^{d_r} \rho_j m_{u,j}^{(k)} \\ &= \sum_{j=2}^{d_r} \rho_j \phi^{-1}\Bigg(1- \\ &\qquad \left[1 - \sum_{i=2}^{d_l} \lambda_i \phi\left(m_{u_0} + (i-1)m_u^{(k-1)}\right)\right]^{(j-1)}\Bigg). \end{aligned} \tag{2.10}$$

For error-free decoding, the recursive equation converges to ∞ as $k \to \infty$, which implies

$$\sum_{i=2}^{d_l} \lambda_i \phi\left(m_{u_0} + (i-1)m_u^{(k-1)}\right) \to 0, \text{as } k \to \infty,$$

because $\phi^{-1}(x) \to \infty$ as $x \to 0$.

For $0 \le s < \infty$ and $0 < r \le 1$, $h_i(s,r)$ and $h(s,r)$ are defined in [8] as

$$\begin{aligned} h_i(s,r) &= \phi\left(s + (i-1)\sum_{j=2}^{d_r} \rho_j \phi^{-1}\left(1-(1-r)^{j-1}\right)\right), \\ h(s,r) &= \sum_{i=2}^{d_l} \lambda_i h_i(s,r), \end{aligned}$$

and (2.10) is equivalently expressed as

$$r_k = h\left(s, r_{k-1}\right), \tag{2.11}$$

where $s = m_{u_0}$ and $r_0 = \phi(m_{u_0})$.

In the recursion, the degree distribution pair $(\lambda(x), \rho(x))$ and s determine the convergence to a fixed point (not necessarily zero). As shown in (2.10) and (2.11), $m_u^{(k)} \to \infty$ iff $r_k \to 0$. Thus, the convergence condition $r_k(s) \to 0$ is satisfied iff

$$h(s,r) < r \quad \forall\ r \in (0, \phi(s)). \tag{2.12}$$

Definition 2 (*definition 2 in [8]*:) *The threshold s^* is the infimum of all s in $\mathbb{R}^+$ such that $r_k(s)$ $(m_u^{(k)}(s))$ converges to 0 (∞) as $k \to \infty$.*

In [8], $\phi(x)$ is approximated as

$$\phi(x) \approx \begin{cases} e^{\alpha x^{\gamma}+\beta} & \text{for} \quad 0 \leq x < 10, \\ \sqrt{\frac{\pi}{x}} e^{-\frac{x}{4}} \left(1 + \frac{1}{14x} - \frac{3}{2x}\right) & \text{for} \quad 10 \leq x, \end{cases}$$

where $\alpha = -0.4527$, $\beta = 0.0218$, and $\gamma = 0.8600$. The approximated function, $\phi(x)$ is accurate enough at all the values of x except around $x = 0$ where $\phi(x) = 1$ but $\phi(x) = e^{\beta} > 1$. While designing LDPC codes over AWGN channels, $\phi(x)$ in $h_j(m, r)$ is computed only at $x \geq m > 0$. Thus, the approximation is good enough. However, we will show that for analyzing punctured LDPC codes we have to compute $\phi(x)$ around $x = 0$. Thus, we modify the approximation as

$$\phi(x) \approx \begin{cases} e^{-ax^2-bx} & \text{for} \quad 0 \leq x < c, \\ e^{\alpha x^{\gamma}+\beta} & \text{for} \quad c \leq x < 10, \\ \sqrt{\frac{\pi}{x}} e^{-\frac{x}{4}} \left(1 + \frac{1}{14x} - \frac{3}{2x}\right) & \text{for} \quad 10 \leq x, \end{cases}$$

where $a = -0.0564$, $b = 0.48560$, and $c = 0.867861$. The parameters, a, b and c are acquired by curve-fitting.

CHAPTER III

LOW-DENSITY PARITY-CHECK CODES OVER GAUSSIAN CHANNELS WITH ERASURES

3.1 Introduction

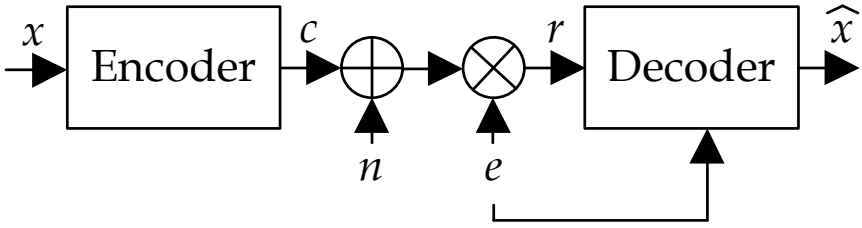
Low-Density Parity-Check (LDPC) codes are due to [14, 15] and renewed interest began as a result of a great success of turbo codes [2] and subsequent rediscovery by MacKay [30], MacKay and Neal [29] and Sipser and Spielman [40]. Since the rediscoveries, much effort has been paid to designing good LDPC codes over various channels such as Binary Erasure Channels (BECs), Binary Symmetric Channels (BSCs) and Additive-White Gaussian Noise (AWGN) channels. Luby *et al.* [27] designed good LDPC codes over BECs and introduced an irregular edge degree distribution. Richardson and Urbanke [38] investigated variations of message densities in message-passing decoders and devised an algorithm called *density evolution* for iteratively computing message densities and the signal-to-noise ratio (SNR) threshold of an LDPC code. Although Chung [6] introduced a simplified density evolution called *discretized density evolution* (DDE), computing thresholds and designing good LDPC codes over most channels, except BECs, require intensive computations. To address the computational difficulty, Chung [8] used the *Gaussian Approximation* (GA) that models the message densities in message-passing decoders as Gaussian distributions for regular LDPC codes and a mixture of Gaussian distributions for irregular LDPC codes. A prerequisite of the approximation is the symmetry condition [37]. At the expense of accuracy, the approximation gives a faster threshold computation, more

insight into the convergence of message-passing decoders, and an analytic expression of the variations of the message densities during iterations. From the design point of view, the approximation makes the design problem a linear optimization that can be easily solved with linear programming.

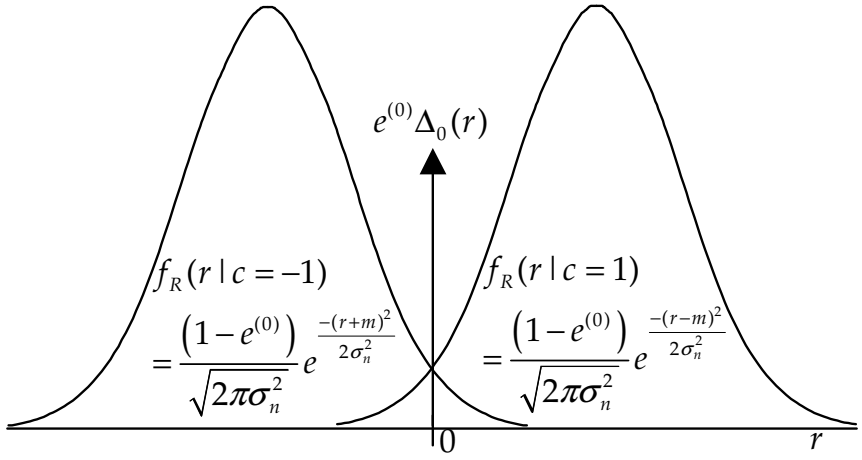
In this paper, we extend Chung's work to include a channel model that is a mixture of Gaussian noise with random erasures, as depicted in Fig. 3.1(a) and 3.1(b). In this model, a binary symbol is corrupted with either added white Gaussian noise or is erased with a probability of $e^{(0)}$. This model, for example, represents a common channel in magnetic recordings where thermal asperities in the system are detected and represented at the decoder as erasures. A typical code must correct a burst of up to 48 bytes over a 512-byte sector. In an optical recording, defects or scratches cause long numbers of bytes (several hundred or thousand) to be erased. The probability density function of the log-likelihood ratio (LLR) of the channel output is

$$\begin{aligned} f(v) &= \frac{1-e^{(0)}}{\sqrt{4\pi m_{u_0}}} e^{-\frac{(v \pm m_{u_0})^2}{4m_{u_0}}} + e^{(0)}\Delta_0(v) \\ &= g^{(0)}(v) + e^{(0)}\Delta_0(v), \end{aligned} \tag{3.1}$$

where $v = \log_e [p(r|c=1)/p(r|c=-1)]$ is the log-likelihood ratio of r, r is the channel output shown in Fig. 3.1(b), $m_{u_0} = E[v|v \neq 0]$, $\mathrm{Var}(v|v \neq 0) = 2m_{u_0}$, $e^{(0)}$ is the random erasure probability, and $\Delta_x(v) = \delta(v-x)$ is a shifted delta function. The probability density function satisfies the symmetry condition [37], $f(v) = f(-v)e^v$. We analyze the message density evolution over the channel model shown in Fig. 3.1(a) and 3.1(b) with some modifications to the Gaussian approximation.



(a)



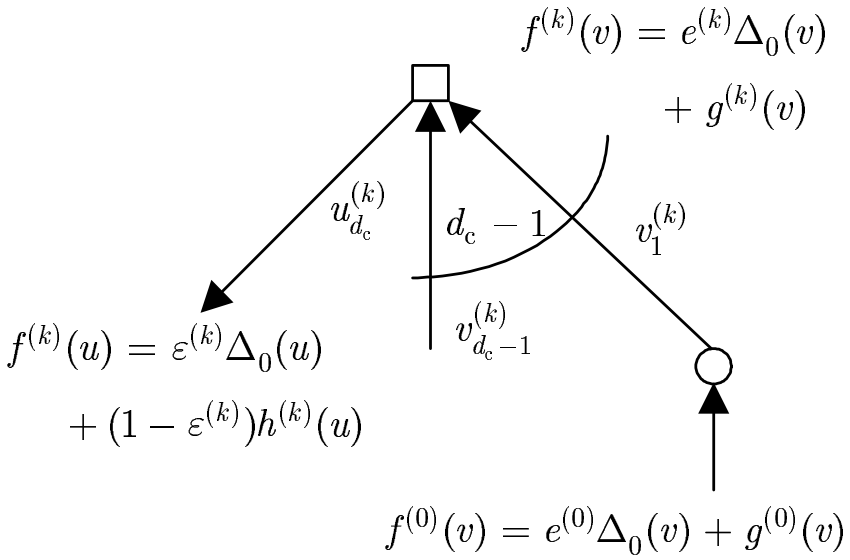
(b)

Figure 3.1: Additive white Gaussian noise with random erasures. (a) x is a message bit in $\{1, 0\}$, c is a coded symbol in {-1, +1}, n is white Gaussian noise ($n \sim \mathcal{N}(0, \sigma_n^2)$), e is in $\{1, 0\}$, $P(e = 0) = e^{(0)}$, $p(e = 1) = (1 - e^{(0)})$, and $\hat{x}$ is a decoded bit in $\{1, 0\}$. (b) r is a received signal and m is the mean of the received signal.

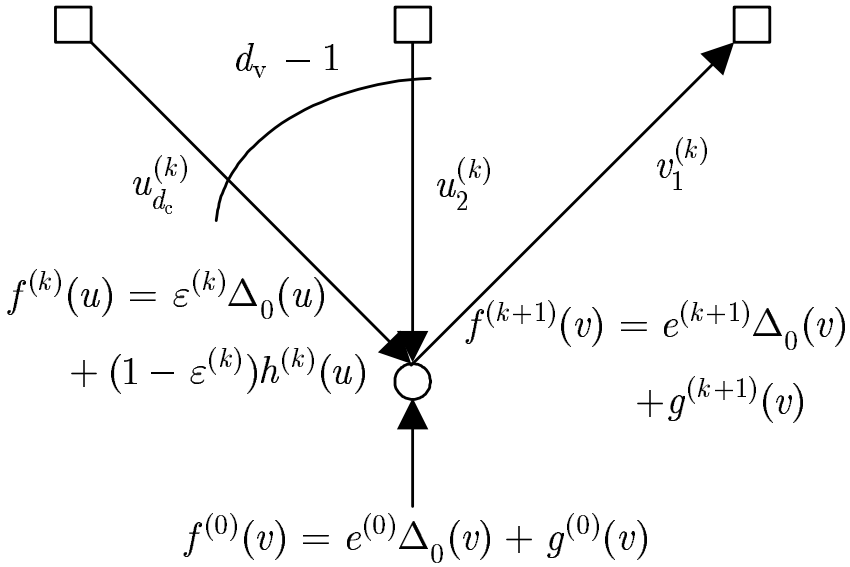
The remainder of the paper is organized as follows. In Section 3.2 we consider regular LDPC codes and define the basic terminology used in the subsequent sections. In Section 3.3 we extend the results of Section 3.2 to irregular LDPC codes. In Section 3.4 we simplify the results of Section 3.3 using the fact that the erasure probability during iterations does not depend on a signal-to-noise ratio. The simplified equation, called a *steady-state equation*, gives us a graphical interpretation of decoder convergence that was originally introduced in [8]. In Section 3.5 we verify our analysis and the channel model by comparing theoretical E_b/N_0 thresholds of regular and irregular LDPC codes with simulation results of actual LDPC codes. We also investigate some practical issues such as maximum number of iterations of message-passing decoders, coded block length and types of erasure patterns (random/block erasures). In Section 3.6 we design an LDPC code for the mixed channel and compare the designed LDPC code with an LDPC code optimized for an AWGN channel in [8]. We conclude our work in Section 3.7.

3.2 Analysis Of Regular LDPC Codes

The message flows between a check and a variable nodes are depicted in Fig. 3.2(a) and 3.2(b), where the square and circle symbols represent check nodes and variable nodes, respectively. The functions, $h^{(k)}(u)$ ($g^{(k)}(v)$) is the continuous part of the probability density of a check (variable) node message. The terms, $\varepsilon^{(k)}$ ($e^{(k)}$) is the probability that a check (variable) node messages is equal to zero. d_{c} (d_{v}) is the number of neighbors of a check (variable) node. $u_j^{(k)}$ ($v_j^{(k)}$) is the message of a check (variable) node emitting through the jth edge (we will drop the edge the index j without loss of generality hereafter). In the sum-product decoding algorithm, if at least one of the variable nodes connected to a check node is erased, the check node has zero LLR as its message. Thus, the probability of $\varepsilon^{(k)}$ is expressed as



(a)



(b)

Figure 3.2: Message flows between a check and a variable nodes. (a) LLR message of a check node at the kth iteration. (b) LLR message of a variable node at the $(k+1)$th iteration.

$$\varepsilon^{(k)} = 1 - \prod_{i=1}^{d_c-1} P(v_i^{(k)} \neq 0) = 1 - \left(1 - e^{(k)}\right)^{d_c-1}. \tag{3.2}$$

If none of $d_c - 1$ variable messages is erased, the probabilistic characteristics of the check message can be approximated by a Gaussian denoted by $h^{(k)}(u)$ in Fig. 3.2. The Gaussian density of a message is determined by one variable, the updated mean value $m_u^{(k)}$ at the kth iteration. The updated mean is determined from the following relation:

$$\begin{aligned} & 1 - \phi\left(m_u^{(k)}\right) \\ & = E\left[\tanh\left[u^{(k)}/2\right] \middle| u^{(k)} \neq 0\right] \\ & = E\left[\tanh[v^{(k)}/2] \middle| v^{(k)} \neq 0\right]^{d_c-1} \\ & = \left[\frac{1}{1-e^{(k)}} \left\langle \tanh\left[v^{(k)}/2\right], g^{(k)}(v)\right\rangle\right]^{d_c-1}, \end{aligned} \tag{3.3}$$

where $\langle f(v), g(v)\rangle = \int_{\mathbb{R}} f(v)g(v)dv$, and $g^{(k)}(v)$ is the probability density of a variable node message given $v^{(k)} \neq 0$. From (3.3), we can express the updated mean value as

$$\begin{aligned} m_u^{(k)} = \phi^{-1}\Bigg(1- \\ \frac{1}{\left(1-e^{(k)}\right)^{d_c-1}} \left\langle \tanh[v^{(k)}/2], g^{(k)}(v)\right\rangle^{d_c-1}\Bigg). \end{aligned} \tag{3.4}$$

To make (3.4) a recursive equation of $m_u^{(k)}$, we need an expression for the probability density of a variable node message. In the log-probability domain, the variable node message is determined by the linear sum of $d_v - 1$ incident check node messages as shown in Fig. 3.2. Thus, the probability that the message of a variable node is zero can be computed as

$$P(v^{(k)} = 0) = e^{(0)}\left(\varepsilon^{(k-1)}\right)^{d_v-1}, \tag{3.5}$$

which is the probability that all incident check messages are zeroes and the variable node is erased. The probability density of a variable message can be factored into three terms as shown in (3.6). The first term gives the probability in (3.5), the second

term is made up of the incident check node messages given the variable node is erased, and the last term is the combination of the check node messages with the unerased received message. The last two terms comprise the continuous part, $g^{(k)}(v)$, used in (3.4).

$$\begin{aligned} f^{(k)}(v) &= e^{(k)}\Delta_0(v) + g^{(k)}(v) \\ &= \underbrace{e^{(k)}\Delta_0(v)}_{\text{Unrecovered erased symbols}} \\ &+ \underbrace{e^{(0)}\left\{\sum_{i=1}^{d_{\mathrm{v}}-1} {}_{(d_{\mathrm{v}}-1)}\chi_i^{(k)}\mathcal{N}\left(im_u^{(k-1)}, 2im_u^{(k-1)}\right)\right\}}_{\text{Erased symbols}} \\ &+ \left(1-e^{(0)}\right)\times \\ &\underbrace{\left\{\sum_{i=0}^{d_{\mathrm{v}}-1} {}_{(d_{\mathrm{v}}-1)}\chi_i^{(k)}\mathcal{N}\left(im_u^{(k-1)}+m_{u_0}, 2im_u^{(k-1)}+2m_{u_0}\right)\right\}}_{\text{Unerased symbols}}, \end{aligned} \tag{3.6}$$

where $\mathcal{N}(m,2m) = \frac{1}{\sqrt{4\pi m}}e^{-\frac{(v-m)^2}{4m}}$, ${}_n\chi_m^{(k)} = {}_n\mathrm{C}_m\left(\varepsilon^{(k-1)}\right)^{n-m}\left(1-\varepsilon^{(k-1)}\right)^m$ and ${}_n\mathrm{C}_m$ is the binomial coefficient.

The continuous part of a variable node message density derived in (3.6) expands (3.4) to

$$\begin{aligned} m_u^{(k)} = \phi^{-1}\Bigg(1 - \frac{1}{\left(1-e^{(k)}\right)^{d_{\mathrm{c}}-1}}\times \\ \Bigg\{1 - e^{(0)}\sum_{i=0}^{d_{\mathrm{v}}-1} {}_{(d_{\mathrm{v}}-1)}\chi_i^{(k)}\phi\left(im_u^{(k-1)}\right) \\ -\left(1-e^{(0)}\right)\sum_{i=0}^{d_{\mathrm{v}}-1} {}_{(d_{\mathrm{v}}-1)}\chi_i^{(k)}\phi\left(im_u^{(k-1)}+m_{u_0}\right)\Bigg\}^{d_{\mathrm{c}}-1}\Bigg), \end{aligned} \tag{3.7}$$

where $\langle\tanh(v/2), \mathcal{N}(x,2x)\rangle = 1-\phi(x)$.

Now, the updated mean becomes a recursive equation for a given random erasure rate $e^{(0)}$.

3.3 Analysis Of Irregular LDPC Codes

To describe irregular LDPC codes, let $\lambda(x) = \sum_{i=2}^{d_l}\lambda_i x^{i-1}$ and $\rho(x) = \sum_{i=2}^{d_r}\rho_i x^{i-1}$ be a pair of generating functions (used in [26]) of the degree distributions for the variable

and check edges, respectively. To include the degree distribution in calculating the probability of $\varepsilon^{(k)}$ and the updated mean $m_u^{(k)}$ that describe the check node message density, we need to modify (3.2) and (3.3) to form (3.8) and (3.9), respectively.

$$\begin{aligned}
\varepsilon_{d_c=s}^{(k)} &= P\left(u^{(k)}=0|d_c=s\right) \\
&= 1-\prod_{i=1}^{s-1} P\left(v_{i=1}^{(k)}\neq 0\right) = 1-\left(1-e^{(k)}\right)^{s-1},
\end{aligned}$$

$$\begin{aligned}
\varepsilon^{(k)} &= \sum_{s=2}^{d_r} P\left(d_c=s\right)\varepsilon_{d_c=s}^{(k)} \\
&= \sum_{s=2}^{d_r} \rho_s\left(1-\left(1-e^{(k)}\right)^{s-1}\right) = 1-\rho\left(1-e^{(k)}\right).
\end{aligned} \tag{3.8}$$

$$\begin{aligned}
&1-\phi\left(m_{u|d_c=s}^{(k)}\right) \\
&= E\left[\tanh\left[u^{(k)}/2\right]\left|u^{(k)}\neq 0, d_c=s\right.\right] \\
&= E\left[\tanh\left[v^{(k)}/2\right]\left|v^{(k)}\neq 0, d_c=s\right.\right]^{s-1},
\end{aligned} \tag{3.9}$$

where $P\left(d_c=s\right)$ is the probability that an edge belongs to a degree s check node. Thus, by making use of (3.4), we compute the updated mean as

$$\begin{aligned}
m_u^{(k)} &= \sum_{s=2}^{d_r} P\left(d_c=s\right) m_{u|d_c=s}^{(k)} \\
&= \sum_{s=2}^{d_r} \rho_s\phi^{-1}\Bigg(1-\frac{1}{\left(1-e^{(k)}\right)^{s-1}}\times \\
&\qquad \left\{\left\langle\tanh\left[v^{(k)}/2\right], g^{(k)}(v)\right\rangle\right\}^{s-1}\Bigg),
\end{aligned} \tag{3.10}$$

where $m_{u|d_c=s}^{(k)} = \phi^{-1}\left(1-E\left[\tanh\left[v^{(k)}/2\right]\left|v^{(k)}\neq 0, d_c=s\right.\right]^{s-1}\right)$ and $E\Big[\tanh\left[v^{(k)}/2\right]\left|v^{(k)}\neq 0, d_c=s\right.\Big]^{s-1} = \frac{1}{\left(1-e^{(k)}\right)^{s-1}}\left\{\left\langle\tanh\left[v^{(k)}/2\right], g^{(k)}(v)\right\rangle\right\}^{s-1}$.

The probability density of a variable node message can be calculated by modifying

(3.5) and (3.6). The expression of $e^{(k)}$ in (3.5) is changed to

$$\begin{aligned} e^{(k)} &= P\left(v^{(k)}=0\right) \\ &= \sum_{j=2}^{d_l} P\left(d_{\mathrm{v}}=j\right) e^{(0)} \left(\varepsilon^{(k-1)}\right)^{j-1} \\ &= e^{(0)} \sum_{j=2}^{d_l} \lambda_j \left(\varepsilon^{(k-1)}\right)^{j-1} \\ &= e^{(0)} \lambda\left(\varepsilon^{(k-1)}\right) \\ &= e^{(0)} \lambda\left(1-\rho\left(1-e^{(k-1)}\right)\right), \end{aligned} \tag{3.11}$$

where $P\left(d_{\mathrm{v}}=j\right)$ is the probability that an edge belongs to a degree j variable node. The recursive erasure probability of a variable node, $e^{(k)}$ is exactly the same as the erasure probability derived by Luby, *et al.* in [27]. There is a threshold $e^{(0)}_{\max}(\lambda,\rho)$ that is the largest real value in $(0,1]$ satisfying $e^{(0)}_{\max}(\lambda,\rho)\lambda(1-\rho(1-e)) < e$ for any $e \in (0, e^{(0)}_{\max}(\lambda,\rho)]$. The threshold does not depend on the SNR but on the code structure specified by a degree distribution pair, ($\lambda(x)$, $\rho(x)$). Thus, as long as the erasure probability ($e^{(0)}$) is less than the threshold ($e^{(0)}_{\max}(\lambda,\rho)$), we can make the probability of an erased symbol as small as possible with the sum-product algorithm. If we increase the SNR of the transmitted signal to infinity, which becomes a pure erasure channel, the probability density of a variable node message becomes $f^{(k)}(v) = e^{(0)}\lambda\left(1-\rho\left(1-e^{(k-1)}\right)\right)\Delta_0(v)$.

The continuous part of the probability density, $g^{(k)}(v)$, can be calculated with proper conditioning on λ_j, and the probability density of a variable node message can

be factorized into three parts as we did in (3.6),

$$
\begin{aligned}
f^{(k)}(v) &= e^{(k)}\Delta_0(v) + g^{(k)}(v) \\
&= e^{(k)}\Delta_0(v) \\
&+ e^{(0)} \left\{ \sum_{j=2}^{d_l} \lambda_j \sum_{i=1}^{j-1} {}_{(j-1)}\chi_i^{(k)} \mathcal{N}\left(im_u^{(k-1)}, 2im_u^{(k-1)}\right) \right\} \\
&+ \left(1 - e^{(0)}\right) \left\{ \sum_{j=2}^{d_l} \lambda_j \times \right. \\
&\left. \sum_{i=0}^{j-1} {}_{(j-1)}\chi_i^{(k)} \mathcal{N}\left(im_u^{(k-1)} + m_{u_0}, 2im_u^{(k-1)} + 2m_{u_0}\right) \right\}.
\end{aligned}
\tag{3.12}
$$

Thus, the continuous part, $g^{(k)}(v)$ expands (3.10) to

$$
\begin{aligned}
m_u^{(k)} = \sum_{s=2}^{d_r} \rho_s \phi^{-1} \Bigg(& 1 - \frac{1}{\left(1 - e^{(k)}\right)^{s-1}} \times \\
& \left[1 - \sum_{j=2}^{d_l} \lambda_j \left\{ e^{(0)} \sum_{i=0}^{j-1} {}_{(j-1)}\chi_i^{(k)} \phi\left(im_u^{(k-1)}\right) \right.\right. \\
& \left.\left. + \left(1 - e^{(0)}\right) \sum_{i=0}^{j-1} {}_{(j-1)}\chi_i^{(k)} \phi\left(im_u^{(k-1)} + m_{u_0}\right) \right\} \right]^{s-1} \Bigg)
\end{aligned}
\tag{3.13}
$$

which becomes a recursive equation of the updated mean, $m_u^{(k)}$. Because we model the message densities as Gaussians, the bit error probability can be computed as a weighted sum of Q functions,

$$
\begin{aligned}
P_e^{(k)} &= e_0 \sum_{j=2}^{d_l} \lambda'_j \sum_{i=0}^{j} {}_j\chi_i^{(k)} Q\left(\sqrt{\frac{im_u^{(k)}}{2}}\right) \\
&+ (1 - e_0) \sum_{j=2}^{d_l} \lambda'_j \sum_{i=0}^{j} {}_j\chi_i^{(k)} Q\left(\sqrt{\frac{im_u^{(k)} + m_{u_0}}{2}}\right)
\end{aligned}
$$

$$
\begin{aligned}
&= \underbrace{e_0 \frac{\sum_{j=2}^{d_l} \left(\lambda'_j \left(\varepsilon^{(k)}\right)^j\right)}{2}}_{\text{Unrecovered erasures}} \\
&\quad + \underbrace{e_0 \sum_{j=2}^{d_l} \lambda'_j \sum_{i=1}^{j} {}_j\chi_i^{(k)} Q\left(\sqrt{\frac{i m_u^{(k)}}{2}}\right)}_{\text{Erased symbols}} \\
&\quad + \underbrace{(1-e_0) \sum_{j=2}^{d_l} \lambda'_j \sum_{i=0}^{j} {}_j\chi_i^{(k)} Q\left(\sqrt{\frac{i m_u^{(k)} + m_{u_0}}{2}}\right)}_{\text{Unerased symbols}},
\end{aligned}
\tag{3.14}
$$

where $\lambda'_j = \dfrac{\lambda_j / j}{\sum_{i=2}^{d_l} \lambda_i / i}$ and $e_0 \sum_{j=2}^{d_l} \lambda'_j \left({}_j\chi_0^{(k)} Q(0)\right) = e_0 \dfrac{\sum_{j=2}^{d_l} \left(\lambda'_j \left(\varepsilon^{(k)}\right)^j\right)}{2}$.

In the error probability, the first term is half of the unrecovered erasure probability at the kth iteration, the second term is the error probability due to the erased symbols, and the last term comes from the unerased symbols.

3.4 Steady-State Recursive Equation

In this section, we provide a simplified *steady-state equation* which gives us a graphical interpretation of the convergence of LDPC codes. As long as $e^{(0)}$ is less than the threshold $e^{(0)}_{\max}(\lambda, \rho)$, $e^{(k)}$ gradually reduces to zero and the recursive equation (3.13) becomes

$$
\begin{aligned}
m_u^{(k)} = \sum_{s=2}^{d_r} \rho_s \phi^{-1} \Bigg(1 - \Bigg[1 - \sum_{j=2}^{d_l} \lambda_j \Big\{ & e^{(0)} \phi\left((j-1) m_u^{(k-1)}\right) \\
& + \left(1 - e^{(0)}\right) \phi\left((j-1) m_u^{(k-1)} + m_{u_0}\right) \Big\} \Bigg]^{s-1} \Bigg)
\end{aligned}
\tag{3.15}
$$

which is a steady-state equation. For the steady-state recursive equation to grow up to infinity, the following inequality must be satisfied

$$\begin{aligned} r &> \sum_{j=2}^{d_l} \lambda_j \left\{ e^{(0)} h_j(0,r) + (1-e^{(0)}) h_j(m_{u_0}, r) \right\} \\ &= e^{(0)} h(0,r) + (1-e^{(0)}) h(m_{u_0}, r) \\ &= H\left(m_{u_0}, e^{(0)}, r\right), \forall\ r \in (0, \phi\left(m_{u_0}\right)) \end{aligned} \tag{3.16}$$

where $h(s,r) = \sum_{j=2}^{d_l} \lambda_j h_j(s,r) = E_{\lambda_j}\left[h_j(s,r)\right]$ and $h_j(s,r) = \phi\Bigg(s + (j-1)\sum_{i=2}^{d_r} \rho_i \phi^{-1}\Big(1-(1-r)^{i-1}\Big)\Bigg)$ as defined in Section 2.4. The inequality is the same as (2.12), if $e^{(0)} = 0$, which corresponds to a AWGN channel. From (3.16), we can notice that the variable node degree distribution λ_j must be optimized to minimize m_{u_0}, while satisfying the inequality for a given erasure probability $e^{(0)} < e^{(0)}_{\max}(\lambda, \rho)$, which becomes a simple linear optimization. The details of the design process will be discussed in Section 3.6.

We explain the steady-state equation with a graphical interpretation by evaluating a 1/2 rate irregular code in [6] whose degree distributions are

$$\begin{aligned} \lambda(x) &= 0.23403x + 0.21242x^2 + 0.14690x^5 + \\ &\qquad 0.10284x^6 + 0.30381x^{19}, \text{ and} \\ \rho(x) &= 0.71875x^7 + 0.28125x^8. \end{aligned} \tag{3.17}$$

In Fig. 3.3, the dotted and the long-dashed gray lines represent $h_i(s,r) - r$ and $h_i(0,r) - r$ for $i = 2, \ldots, 20$ from the top to the bottom, respectively. By averaging $h_i(s,r) - r$ and $h_i(0,r) - r$ with λ_j, we can compute $h(s,r) - r$ and $h(0,r) - r$ denoted by the black dotted and the long-dashed lines, respectively. Finally, $H(s, e^{(0)}, r) - r$ is calculated by averaging $h(s,r) - r$ and $h(0,r) - r$ by $(1 - e^{(0)})$ and $e^{(0)}$, respectively. To satisfy the inequality in (3.16), $H(s, e^{(0)}, r) - r$ must be less than zero, $\forall\, r \in (0, \phi(s)]$. Therefore, over the mixed channel, the threshold $m^M_{u_0}(e^{(0)})$ is the minimum of m_{u_0} such that for a given erasure probability $e^{(0)} < e^{(0)}_{\max}(\lambda, \rho)$, if $s \geq m_{u_0}$, then $H(s, e^{(0)}, r) - r < 0$, $\forall\, r \in (0, \phi(s)]$. Over an AWGN channel, the threshold, $m^G_{u_0}$ is the minimum of m_{u_0}

such that if $s \geq m_{u_0}$ then $H(s, 0, r) - r = h(s, r) - r < 0, \forall\, r \in (0, \phi(s)]$, which is the same as definition 2 in Section 2.4. Fig. 3.4 shows a magnification of $H(s, e^{(0)}, r) - r$ when $e^{(0)} = 0.38$ and $s = 0.6320891$. From the curve, we can see that $H(s, e^{(0)}, r) - r$ barely avoids touching zero. If it does reach zero, the point of intersection becomes a fixed point of the recursive equation.

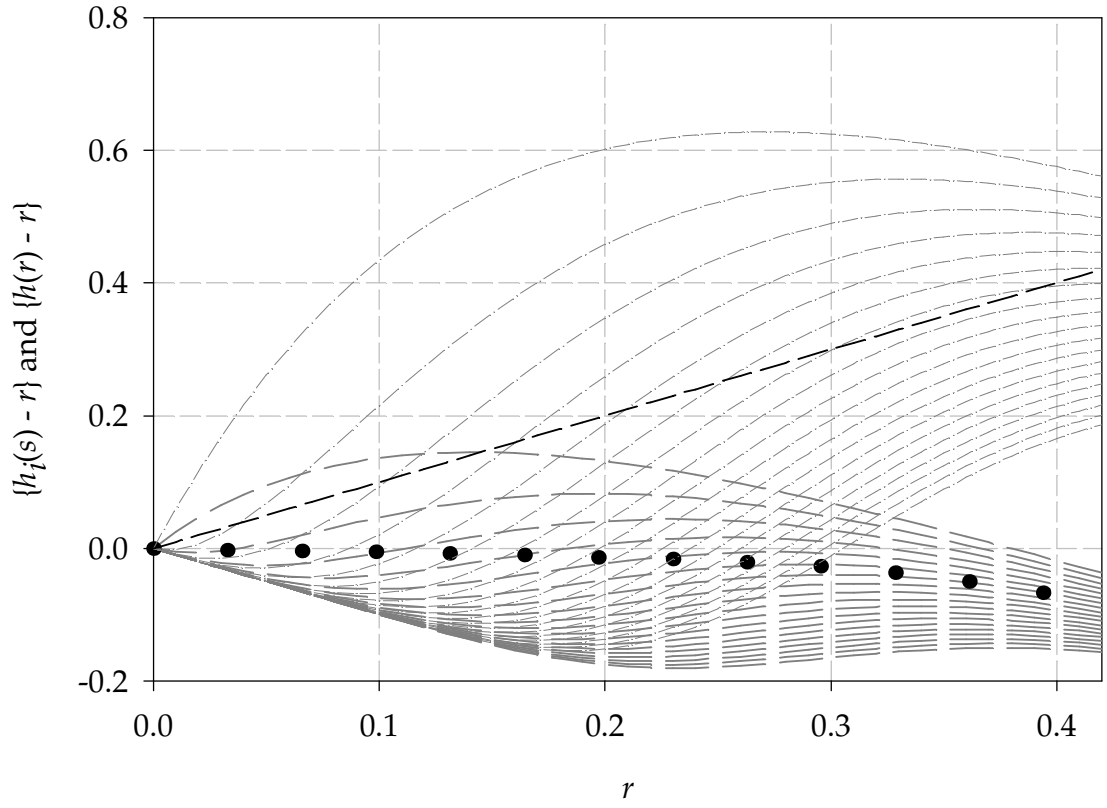


Figure 3.3: $\{h_i(s,r) - r\}$ for $i = 2, \ldots, 20$ (top to bottom) , and $\{h(s,r) - r\}$ for $s = 0$ and 6.320891.

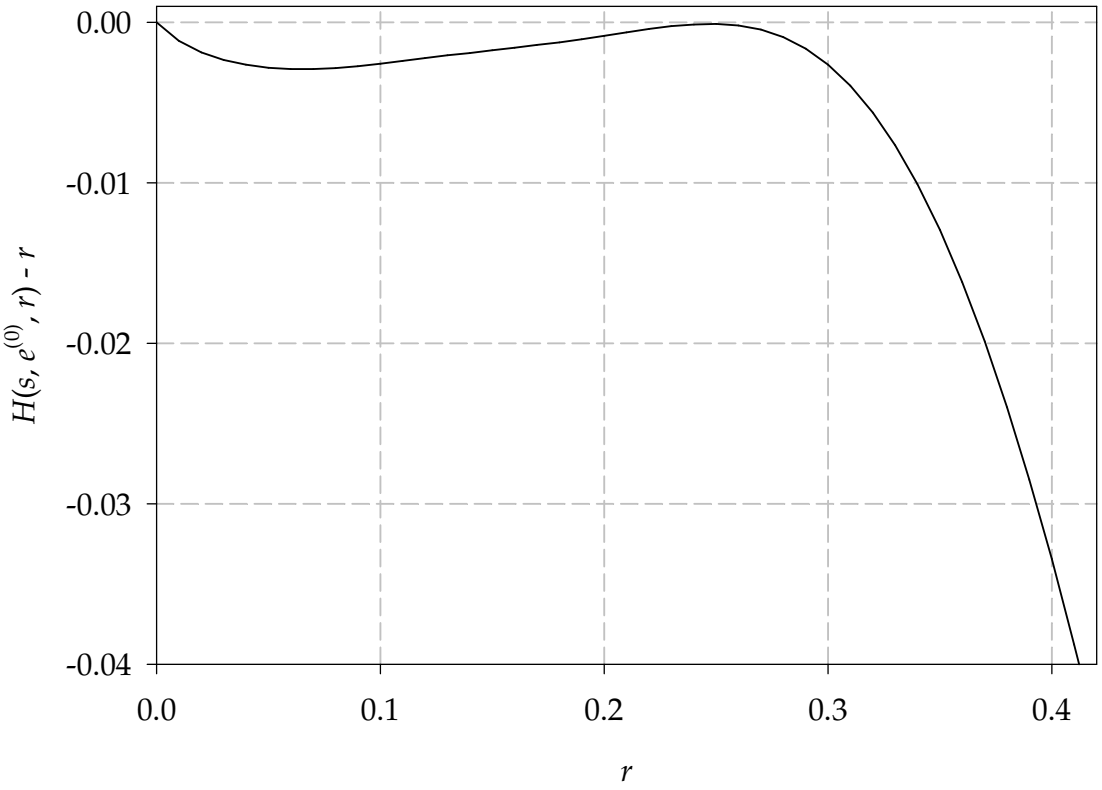


Figure 3.4: A magnified view of $H(s, e^{(0)}, r) - r$ when $s = 6.320891$ and $e^{(0)} = 0.38$.

3.5 Performance Prediction

In this section we verify our analysis and the channel model by comparing theoretical E_b/N_0 thresholds of regular and irregular LDPC codes with simulation results of actual LDPC codes. We also investigate some practical issues concerning small maximum number of iterations of the message-passing decoders, types of erasure patterns and short block lengths.

In magnetic storage, a code block length of 4096 and coding rates of 4/5, 8/9 and 16/17 are common. In Fig. 3.5 we compare simulation results of corresponding regular LDPC codes for a bit-error rate (BER) of 10^{-4} with the thresholds predicted by Gaussian Approximation (GA). In the simulation, we limit the maximum number of iterations of the message-passing decoder to 25. The computed theoretical thresholds fairly predict asymptotic performances of the LDPC codes, which confirms our analysis of regular LDPC codes over the mixed channel.

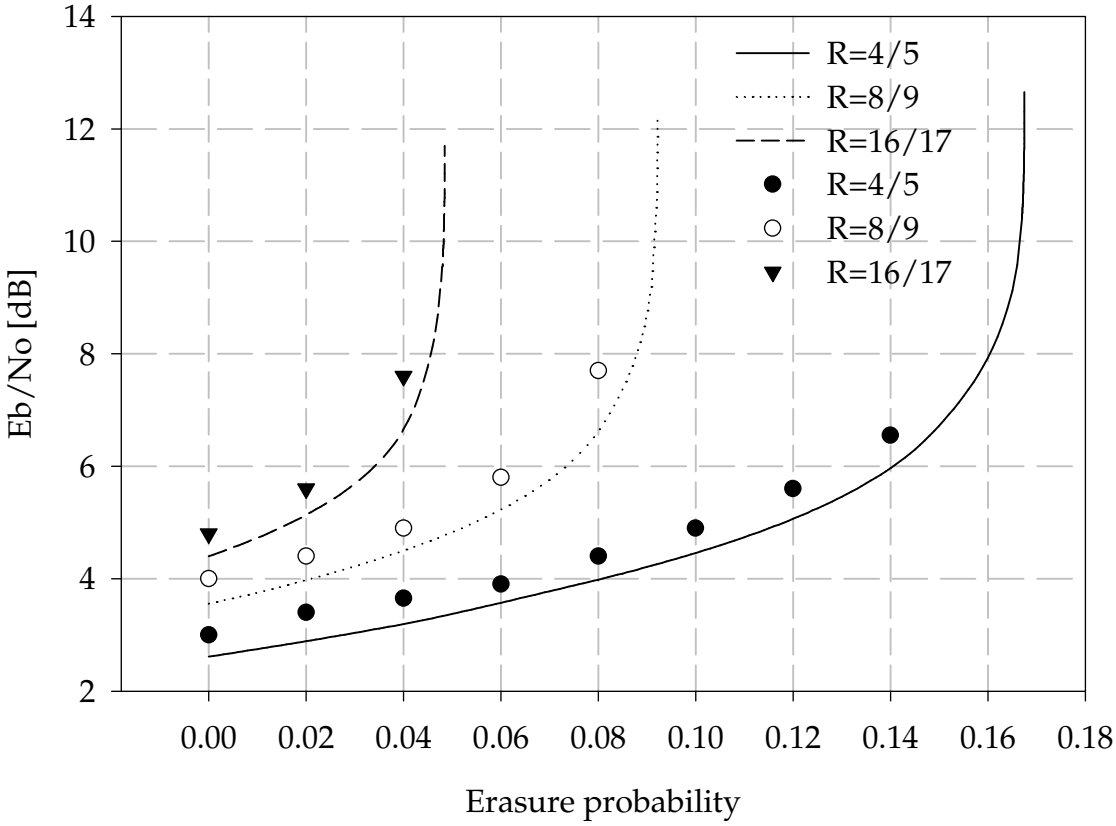


Figure 3.5: Comparison of E_b/N_0 variations of regular LDPC codes with erasures at the coding rates of 4/5, 8/9, and 16/17 which are made of 3 ($\lambda_3 = 1$) nonzero terms in each column and 15 ($\rho_{15} = 1$), 27 ($\rho_{27} = 1$), and 51 ($\rho_{51} = 1$) nonzero terms in each row of the parity check matrices, respectively; dots represent simulation results for a code length of 4096 and a bit-error rate of 10^{-4}, and lines are corresponding thresholds predicted with GA.

Although regular LDPC codes may be a more practical choice, irregular LDPC codes give us more freedom in searching for capacity-approaching LDPC codes. In Fig. 3.6 we evaluate BER performances of an irregular LDPC code. The code is of length 131072 and is implemented with the degree distribution pair in (3.17). The analysis with GA fairly predicts the thresholds of the LDPC code over the mixed channel. In Fig. 3.7 we depict thresholds predicted by Gaussian Approxmation (GA) and Discretized Density Evolution (DDE) [6], which are compared with the required E_b/N_0's for a BER of 10^{-4} shown in Fig. 3.6. To see the effect of the maximum number of iterations and erasure patterns, we did the same simulation with two different values for the maximum number of iterations (25 and 200) and types of erasure patterns (random and block erasures). The thresholds predicted by GA and DDE have a 0.2dB discrepancy, which was observed for the AWGN channel in [8]. In Fig. 3.8 we depict a magnified view of the gaps between the thresholds and the capacity of the mixed channel, $C_{\mathrm{BPSK}}(1-e^{(0)})$. The threshold from DDE is less than 0.2 dB from the capacity as long as the erasure probability is less than 0.2. Thus, good (capacity-approaching) LDPC codes over AWGN channels are still good over the mixed channels up to a moderate erasure probability. However, at a high erasure probability there is a more distinct gap from the capacity. We can design an LDPC code optimized for a high erasure probability or for a small average gap from the capacity over several erasure probabilities. Although the optimized LDPC code has better thresholds at high erasure probabilities than those of an LDPC code for an AWGN channel, the LDPC code will suffer some loss in performance at low erasure probabilities.

Figs. 3.7 and 3.8 also show the effects of the maximum iterations and the types of erasure patterns (random and block erasures). In both erasure cases, we erase a fixed number of symbols. In the random erasure case, the positions of erased symbols are chosen randomly. In the block erasure case, we randomly choose the beginning

of an erasure block from which we erase consecutive symbols. The block and random erasure cases (unfilled circles and triangles) show almost the same performance with a large number of iterations (200 iterations). We see performance degradation (filled circles and triangles) in both cases due to a smaller number of iterations (25 iterations). However, the degradation is more severe in the block erasure case. In Fig. 3.8 the additional E_b/N_0's (the additional gaps form the capacity) in the random erasure case do not depend on the erasure probabilities. That is, at each erasure probability, the additional E_b/N_0 is almost the same. In the block erasure case, more additional E_b/N_0 is required for a higher erasure probability. Because block erasures can be efficiently converted to random erasures with a proper interleaver, the results in Section 3.2, 3.3 and 3.4 are still useful in the block erasure case.

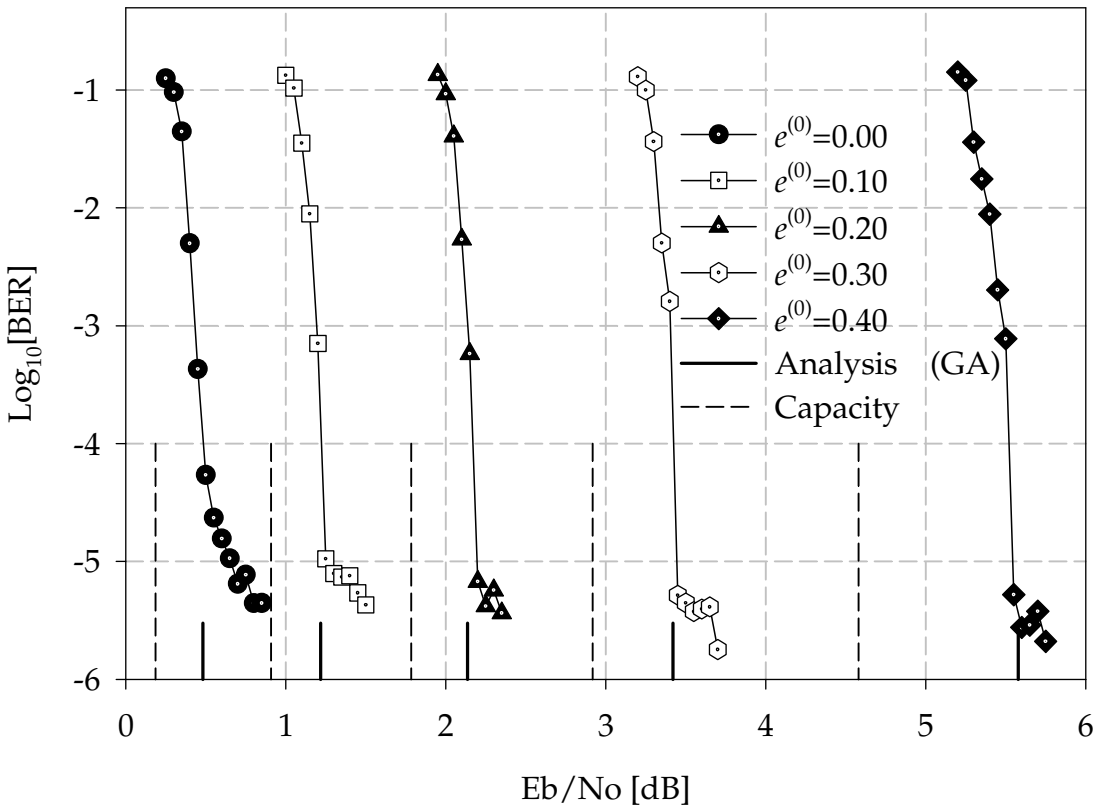


Figure 3.6: Bit error rates of an irregular LDPC code having a code block length of 131702 bits, a coding rate of 1/2, maximum iterations of 200 and edge degree distributions, $\lambda(x) = 0.23403x + 0.21242x^2 + 0.14690x^5 + 0.10284x^6 + 0.30381x^{19}$ and $\rho(x) = 0.71875x^7 + 0.28125x^8$.

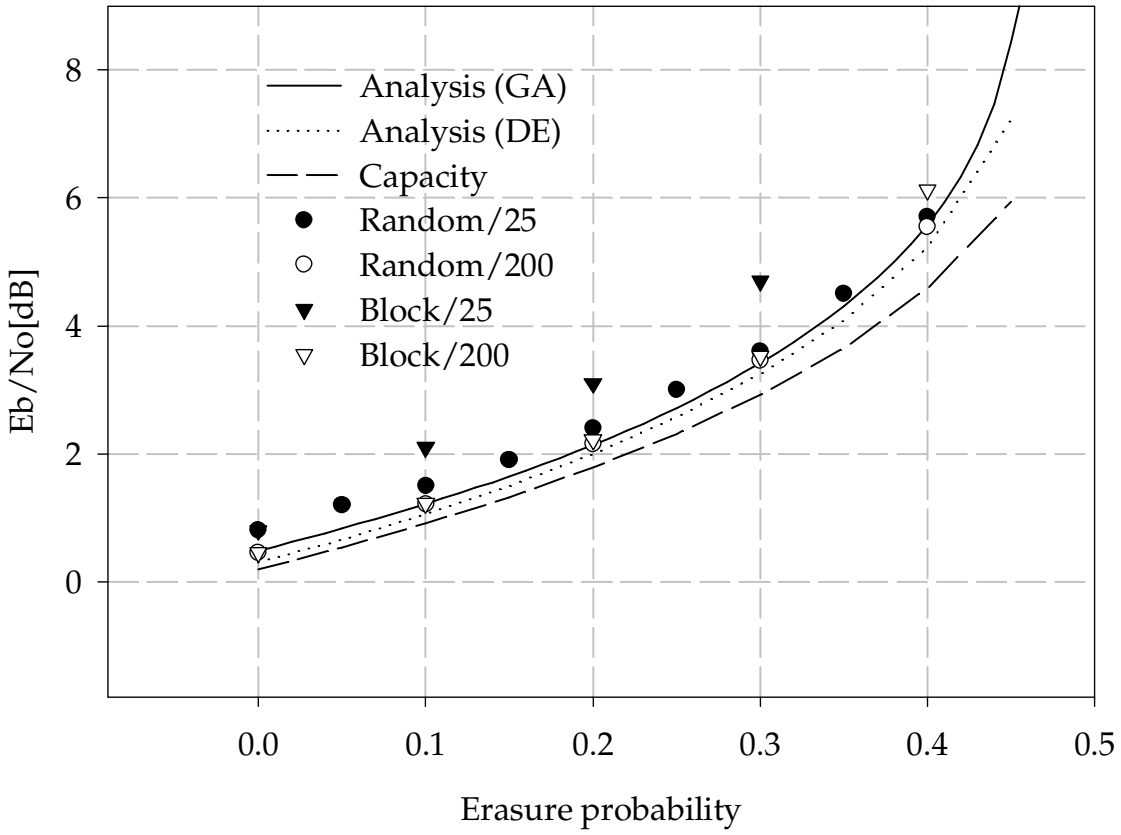


Figure 3.7: E_b/N_0 variations regarding erasure probabilities; filled and unfilled circles and triangles represent the results of an irregular LDPC code having code block length of 131702 bits, a coding rate of 1/2, and maximum iterations of 25 and 200, respectively. Edge degree distributions are $\lambda(x) = 0.23403x + 0.21242x^2 + 0.14690x^5 + 0.10284x^6 + 0.30381x^{19}$ and $\rho(x) = 0.71875x^7 + 0.28125x^8$.

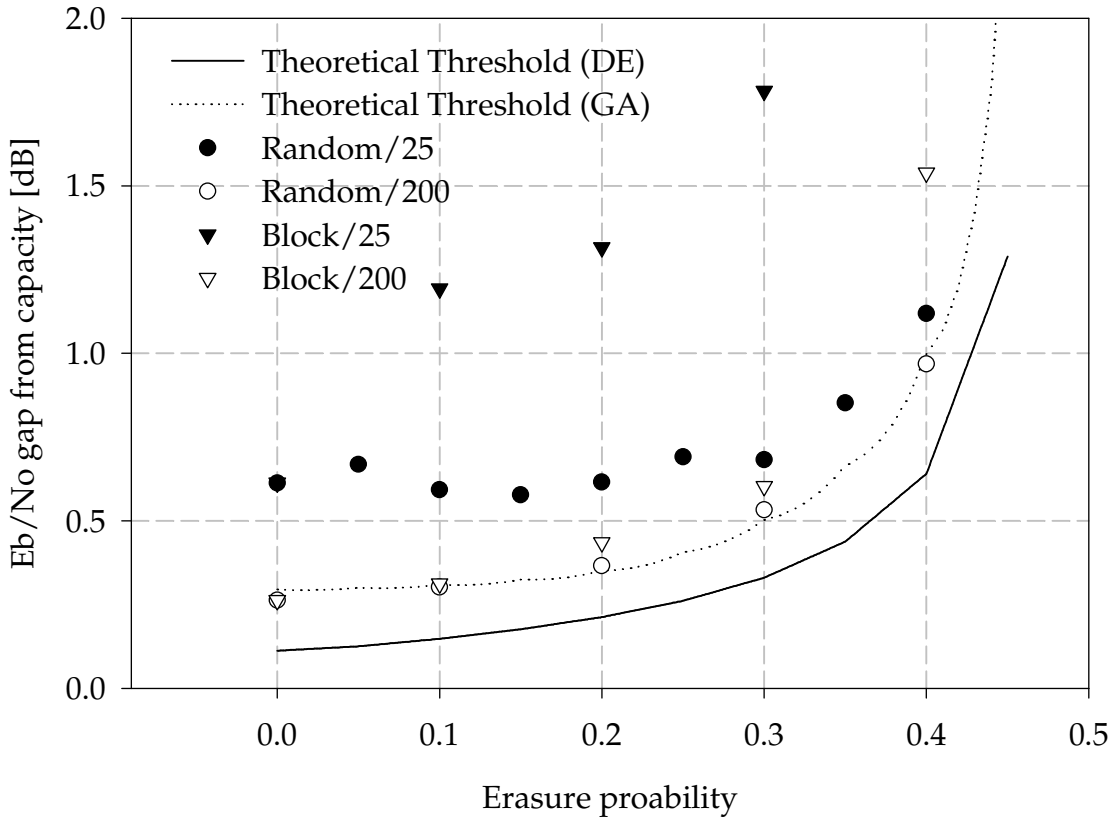


Figure 3.8: E_b/N_0 gaps from the capacity;filled and unfilled circles and triangles represent the results of an irregular LDPC code having code block length of 131702 bits, a coding rate of 1/2, and maximum iterations of 25 and 200, respectively. Edge degree distributions are $\lambda(x) = 0.23403x + 0.21242x^2 + 0.14690x^5 + 0.10284x^6 + 0.30381x^{19}$ and $\rho(x) = 0.71875x^7 + 0.28125x^8$.

We evaluate E_b/N_0 performances of the LDPC code used in Figs. 3.7 and 3.8 with several different block lengths (131072, 16384 and 4096), erasure patterns (random and block erasures), and maximum iterations. In Fig. 3.9 we depict the required E_b/N_0's for a BER of 10^{-4} with a fixed maximum number of iterations, 200. At each erasure probability, the block and random erasure patterns have the same performances. Fig. 3.10 shows the variations of required E_b/N_0's with different maximum iterations at an erasure probability of 0.1. The results show that the performance variation decreases as the number of maximum iterations increases. Block and random erasure patterns have the same required E_b/N_0 as the maximum number of iterations becomes large enough (200 in our simulations). Figs. 3.9 and 3.10 have a similar trend to Figs. 3.7 and 3.8. Thus, either a large number of iterations or a proper interleaver can remove the performance variations due to the erasure patterns. Yang and Ryan [46] designed LDPC codes based on our results and showed that better designed LDPC codes can also efficiently avoid the performance variation between random and block erasure patterns.

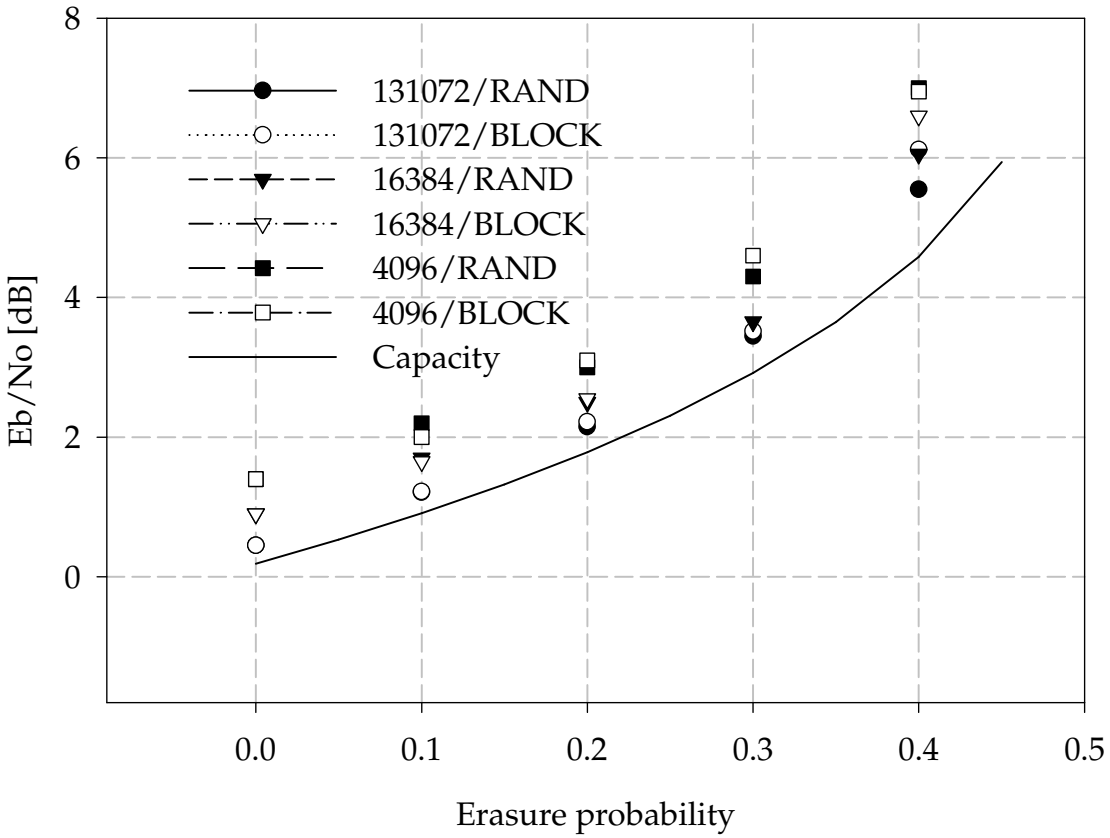


Figure 3.9: E_b/N_0 variations of LDPC codes for a bit-error rate of 10^{-4} with the erasure probabilities between 0 and 0.4 by a 0.1 step, a maximum number of iterations of 200, and the random and block erasure patterns.

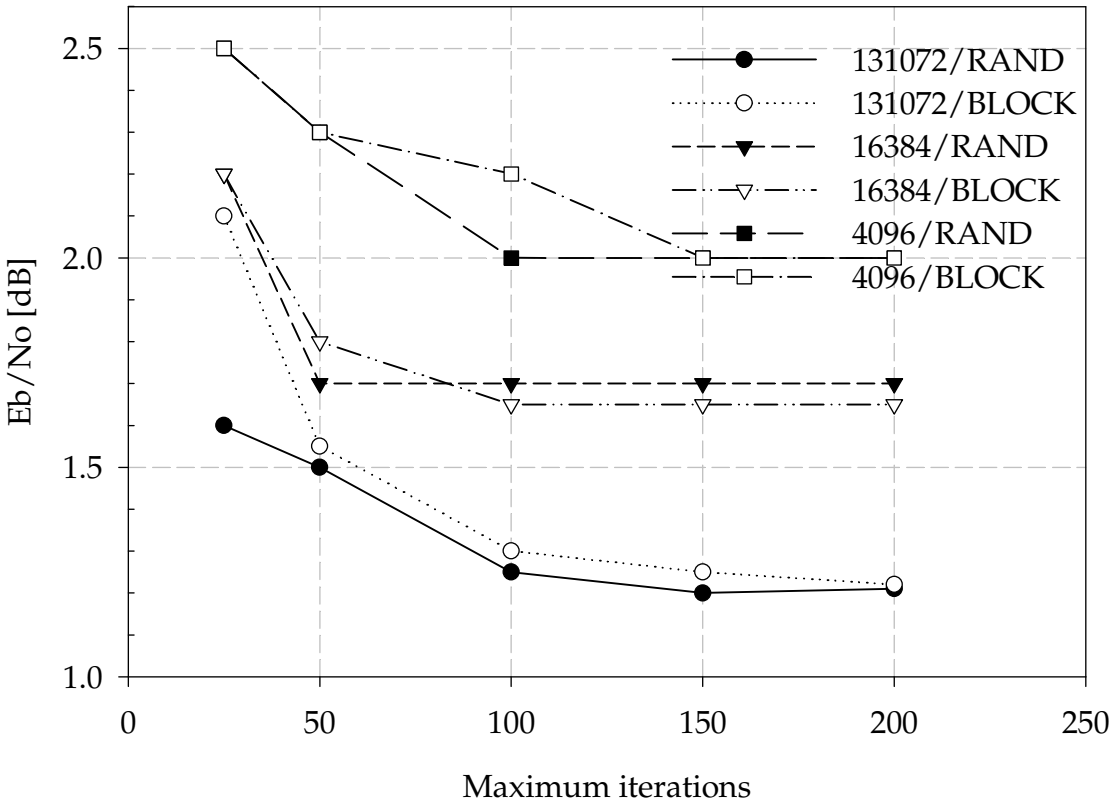


Figure 3.10: E_b/N_0 variations of LDPC codes for a bit-error rate of 10^{-4} with different maximum iterations (25, 50, 100, 150, and 200), erasure probability of 0.1, and random and block erasure patterns.

3.6 Design Of An LDPC code For The Mixed Channel

In this section, we discuss code optimization with respect to left and right degree distributions to minimize the following metric,

$$\min_{\arg\{\lambda(x),\rho(x)\}} M = \sum_{j=0}^{L} w_j \{E(\lambda(x), \rho(x), e_j) - K(e_j, r)\}, \tag{3.18}$$

$$\text{given } r = 1 - \frac{\sum_{j=2}^{d_r} \rho_j / j}{\sum_{j=2}^{d_l} \lambda_j / j},$$

where $E(\cdot)$ is the E_b/N_0 threshold computed with GA given the degree distribution pair $(\lambda(x), \rho(x))$, $K(\cdot)$ is E_b/N_0 of the capacity for a coding rate of r, e_j's are the erasure probabilities at which $E(\cdot)$ and $K(\cdot)$ are evaluated, and w_j's are positive real weighting factors.

For an LDPC code on an AWGN channel, the weight factors become $w_0 = 1$ and $w_j = 0$ for all $j \neq 0$, and $e_0 = 0$. The optimization on the left degree distribution can be done through a linear optimization as described in Section 3.4 for a given right edge degree distribution that is jointly optimized through the differential evolution technique in [36]. It is also possible to design an LDPC code with the differential evolution technique on the left and right degree distributions with the recursive equation (3.14). We state the latter design procedure as

1. [Initialization] Set $0 < \min E_b/N_0 < \max E_b/N_0$, $1 \ll N_{\max}$, $0 < \min P_e \ll 1$, $0 < \min \delta P_e \ll 1$, and $n = 0$

2. $n = n + 1$

3. [Differential Evolution] Get a trial vector composed of a pair of distributions $(\lambda_n(x), \rho_n(x))$ from the differential evolution whose coding rate is fixed

4. [Compute Thresholds] Compute the threshold, $E\left(\lambda_n(x), \rho_n(x), e_j\right)$ at each e_j

   a) Set $1 \ll K_{\max}$, $0 < \delta E \ll 1$, $E_{\max} = \max E_b/N_0$, and $E_{\min} = \min E_b/N_0$

   b) $k = 0$, and $P_e^{(0)} = e_0/2 + (1 - e_0)Q\left(\sqrt{m_{u_0}/2}\right)$

   c) $E\left(\lambda_n(x), \rho_n(x), e_j\right) = (E_{\max} + E_{\min})/2$

   d) If $(E_{\max} - E_{\min}) < \delta E$ go to 5

   e) $k = k + 1$

   f) Compute $\varepsilon^{(k)}$ in (3.8) and store $\varepsilon^{(k)}$ for computing $\varepsilon^{(k+1)}$ at the next iteration

   g) Compute $m_u^{(k)}$ in (3.13) and store $m_u^{(k)}$for computing $m_u^{(k+1)}$ at the next iteration

   h) Compute $P_e^{(k)}$ defined in (3.14) and $\delta P_e^{(k)} = (P_e^{(k-1)} - P_e^{(k)})/P_e^{(k-1)}$ with $\varepsilon^{(k)}$ and $m_u^{(k)}$

   i) If $P_e^{(k)} < \min P_e$ then $E_{\max} = E\left(\lambda_n(x), \rho_n(x), e_j\right)$ and go to b)

   j) If $K_{\max} < k$ or $\delta P_e^{(k)} < \min \delta P_e$ then $E_{\min} = E\left(\lambda_n(x), \rho_n(x), e_j\right)$ and go to b)

   k) Go to e)

5. [Compute Metric] Compute the metric in (3.18) with $E\left(\lambda_n(x), \rho_n(x), e_j\right)$'s for $0 \leq j \leq L$ and return the metric to the differential evolution routine that updates the best metric and the degree distribution pair

6. If $n < N_{\max}$ then go to 2

7. [Termination] Get the best degree distribution, $(\lambda^*(x), \rho^*(x))$ from the differential evolution routine and stop.

The design procedure is a combination of the differential evolution and the binary search algorithm. In the design procedure, we search for the best degree distribution pair $(\lambda^*(x), \rho^*(x))$ that minimizes the metric defined in (3.18). The equality in each step is an assignment operator instead of a boolean operator. That is, the left variable will be the same as the right variable. In step 1 we define the initial values, where $\max E_b/N_0$ and $\min E_b/N_0$ are the upper and lower limits of the E_b/N_0 threshold at each erasure probability (e_j), $N_{\max}$ is the maximum number of iterations of this design procedure, and $\min P_e$ and $\min \delta P_e$ are the minimum bit-error probability and the minimum bit-error probability decrease, respectively, which are explained in step 4. In step 3 the differential evolution routine returns a randomly generated edge degree distribution pair, $(\lambda_n(x), \rho_n(x))$. In step 4 the threshold at each erasure probability (e_j) is evaluated with the binary search algorithm. In the binary search the threshold of an LDPC code is presumed to be between $E_{\max}$ and $E_{\min}$. In a), $E_{\max}$ and $E_{\min}$ are set to $\max E_b/N_0$ and $\min E_b/N_0$, respectively. $E(\lambda_n(x), \rho_n(x), e_j)$ will be always the middle of $E_{\max}$ and $E_{\min}$. In d), if the difference between $E_{\max}$ and $E_{\min}$ is less than $\delta E \ll 1$, which means the threshold is within $\delta E/2$ from the true threshold, the binary search terminates. From e) to h), the bit error-probability at the kth iteration, $P_e^{(k)}$, and the decrease of the bit-error probability, $\delta P_e^{(k)}$, are evaluated. In i), if the bit-error probability is less than $\min P_e$ that is regarded as zero probability, the threshold is less than or equal to the current threshold. Thus, $E_{\max} = E(\lambda_n(x), \rho_n(x), e_j)$ and a search for the thresholds between $E_{\max}$ and $E_{\min}$ is performed again. In j), if $K_{\max} < k$ (too many iterations) or $\delta P_e < \min \delta P_e$, which means the iterative bit-error probability falls into a fixed point, the threshold must be larger than the current threshold, $E(\lambda_n(x), \rho_n(x), e_j)$. In step 5 with the thresholds found through the binary search, the metric defined in (3.18) is passed to the differential evolution routine that makes a new random vector in step 3. The steps from 3 to 5 will be iterated $N_{\max}$ times, and we get the best results $(\lambda^*(x), \rho^*(x))$.

One can design a code that has been optimized for a specific erasure probability. We are more interested in a single code that is optimized across a range of erasure probabilities. Again, this is common in recording when the code is designed to handle a maximum number of erasures. We consider an LDPC code which has a coding rate of 1/2 and is optimized with uniform weighting factors. The metric in (3.18) is evaluated at every erasure probability between 0 and 0.35 with a 0.05 step. In Fig. 3.11 we compare between a well-designed LDPC code with GA for an AWGN channel in [8] and an LDPC code designed for the mixed channel, where the thresholds are computed with GA and DDE. The edge degree distribution pairs of the LDPC code in [8] and the designed one are listed in Table 3.1. The designed LDPC code shows better thresholds if the erasure probability is greater than 0.1 in both the GA and DDE cases. The thresholds from DDE with erasure probability less than 0.1 are poorer than the LDPC code designed for the AWGN channel, which can be either mitigated or avoided with proper weighting factors.

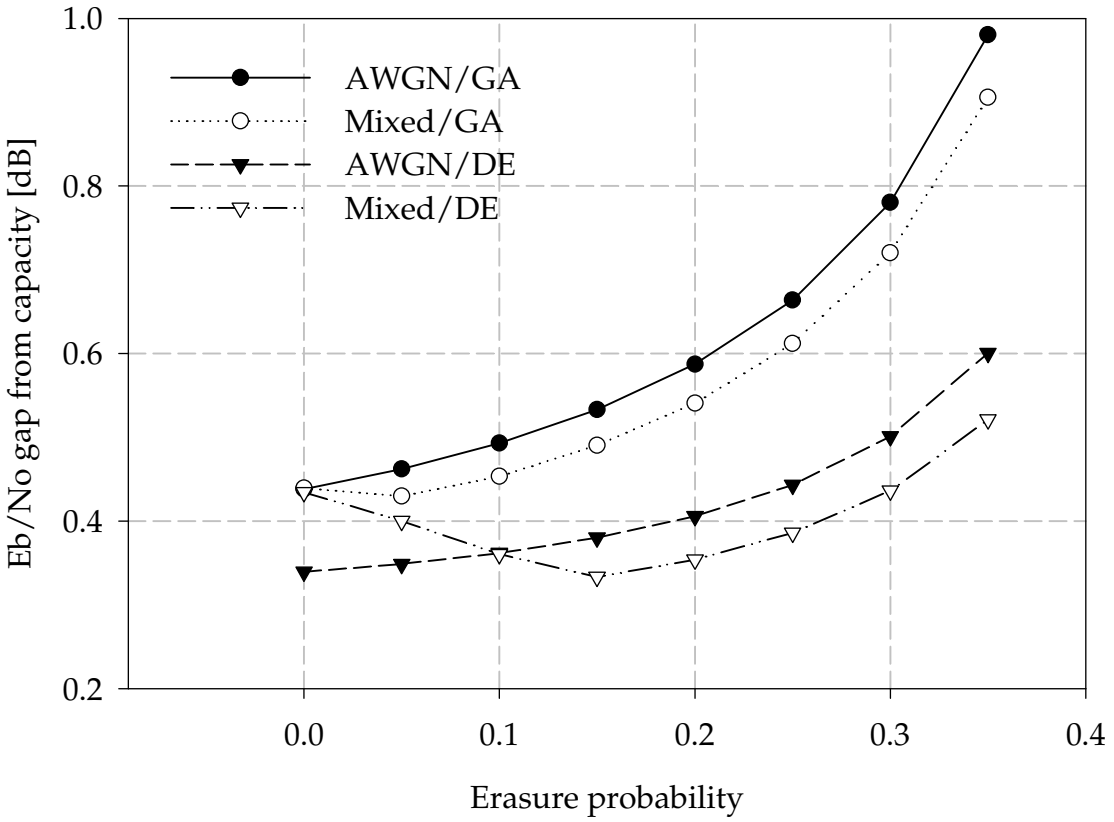


Figure 3.11: E_b/N_0 gap from the capacity; LDPC code designed for AWGN channel having a degree distribution of $\lambda(x) = 0.30780x + 0.27287x^2 + 0.41933x^6$ and $\rho(x) = 0.4x^5 + 0.6x^6$ and LDPC code designed over the mixed channel having degree distribution $\lambda(x) = 0.33175x + 0.24120x^2 + 0.42705x^6$ and $\rho(x) = 0.45290x^5 + 0.54710x^6$.

Table 3.1: Edge degree distribution pairs of an LDPC code designed for an AWGN channel $(\lambda(x),\ \rho(x))$ and one designed for the mixed channel $(\lambda^*(x),\ \rho^*(x))$.

Degree	$\lambda(x)$	$\lambda^*(x)$
2	0.30780	0.33175
3	0.27287	0.24120
7	0.41933	0.42705
Degree	$\rho(x)$	$\rho^*(x)$
6	0.40000	0.45290
7	0.60000	0.54710

3.7 Conclusions

The Gaussian approximation is extended to include the mixture of white Gaussian noise with erasures, and the analytic results not only provide a way to predict asymptotic performance of LDPC codes but also help us design good LDPC codes over the mixed channel. The analysis tells us that the variation of the erasure probability during iterations does not depend on the received signal power but only on the structure of LDPC codes, which can be defined with edge degree distributions. Thus, as long as the erasure probability is less than a threshold of an LDPC code over BECs, we can assume that the erasure probability monotonically decreases to zero, which gives us a simple recursive equation describing message densities, called a steady-state equation. This simplified equation leads us to a design rule of good LDPC codes over the mixed channel and a graphical interpretation of the convergence of LDPC codes over the mixed channel.

To show the validity of the analytic results and the channel model, we simulated an LDPC code having a code block length of 131072 and a coding rate of 1/2. We also computed the E_b/N_0 thresholds of the LDPC code using the analytic results. The comparison between the simulation results and the thresholds showed that the

prediction from the theoretical results was accurate enough.

We have found that well-designed LDPC codes for AWGN channels are still good over the mixed channel. In the simulation, we had less than 0.1 dB E_b/N_0 loss up to an erasure probability of 0.2. We also investigated the performance variations due to types of erasure patterns with different block lengths, values for maximum iterations. The simulation results allowed us to conclude that to mitigate the performance variations, we need a large number of iterations or a proper interleaver before the decoder.

Although good LDPC codes are also good over the mixed channel, we can, in the average sense, improve LDPC codes over several erasure probabilities. We optimized an LDPC code for the mixed channel with a design rule from the analytic results to achieve a better average metric. The designed code shows better thresholds at erasure probabilities larger than 0.1 by 0.1dB in GA and DDE. From a practical point of view, we believe our work would be helpful to predict thresholds of LDPC codes over the mixed channel, especially for magnetic and optical storage applications. From a theoretical point of view, it also paves the way for a new study of LDPC codes over the mixed channel.

CHAPTER IV

RATE COMPATIBLE PUNCTURED LOW-DENSITY PARITY-CHECK CODES

4.1 Introduction

On time varying channels, a common error control strategy is to adapt the coding rate according to available channel state information (CSI). An effective way to realize this coding strategy is to use a single code and puncture it in a rate-compatible fashion, a so-called rate-compatible punctured code (RCPC) [31, 4, 21]. In such an approach, the transmitter systematically punctures parity bits in a coded block, and the locations of punctured symbols are known to the receiver/receivers. Because the decoder for the lowest coding rate (the base code) is compatible with the other higher coding rates, RCPC needs no additional complexity for the coding rate adaptability. Moreover, RCPC permits one to transmit redundancies progressively in conjunction with automatic repeat request (ARQ) [31, 28].

In this paper we apply this idea to low-density parity-check (LDPC) codes, namely one would like to have a single LDPC code which, when punctured in a rate-compatible way, remains good across a range of punctured rates. We present a way to puncture LDPC codes which does not disturb the optimality of the base code, and where the resulting punctured codes maintain threshold optimality across a range of rates. We focus mainly on asymptotic thresholds of the punctured LDPC codes instead of practical issues such as code performance with a short block length, number of iterations to achieve a saturated performance and finite precision effects due to quantization.

LDPC codes can be defined by an $r \times n$ sparse parity-check matrix, $\mathbf{H}$ that can be

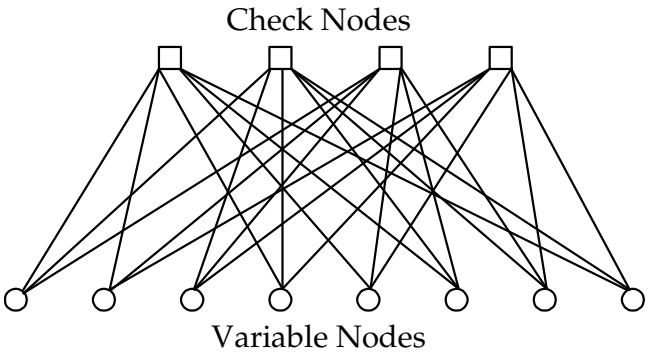


Figure 4.1: A bipartite graph of (3, 6) regular code.

graphically represented by a bipartite graph [41] depicted in Fig. 4.1. On the graph, each column and row of the sparse parity-check matrix are denoted as a *variable* node and a *check* node and represent a received symbol and a parity check, respectively.

While the parity-check matrix and bipartite graph specify an instance of an LDPC code, an ensemble of LDPC codes can be described with a degree distribution pair $(\lambda(x), \rho(x))$ [27]. $\lambda(x) = \sum_{i=2}^{d_l} \lambda_i x^{i-1}$ ($\rho(x) = \sum_{i=2}^{d_r} \rho_i x^{i-1}$) is a polynomial whose coefficients are nonnegative real numbers, the sum of the coefficients is equal to 1, and λ_i (ρ_i) is the fraction of edges belonging to variable (check) nodes with i edges (or a degree of i), and d_l (d_r) is the maximum variable (check) node degree. Thus, we can have many different parity-check matrices which comply with a degree distribution pair, and either a parity-check matrix or bipartite graph is a realization of a degree distribution pair. The coding rate of LDPC codes specified by a degree distribution pair, $(\lambda(x), \rho(x))$ is computed as (see [37])

$$r(\lambda, \rho) = 1 - \frac{\int_0^1 \rho(x)\mathrm{d}x}{\int_0^1 \lambda(x)\mathrm{d}x} = 1 - \frac{\sum_{j=2}^{d_r} \rho_j / j}{\sum_{j=2}^{d_l} \lambda_j / j}.$$

The degree distribution from an edge perspective can be converted to a node perspective with the following relations:

$$\lambda'_j = \frac{\lambda_j/j}{\sum_{i=2}^{d_l} \lambda_i/i} \Leftrightarrow \lambda_j = \frac{j\lambda'_j}{\sum_{i=2}^{d_l} i\lambda'_i}, \text{ and}$$
$$\rho'_j = \frac{\rho_j/j}{\sum_{i=2}^{d_r} \rho_i/i} \Leftrightarrow \rho_j = \frac{j\rho'_j}{\sum_{i=2}^{d_r} i\rho'_i}, \tag{4.1}$$

where λ'_j (ρ'_j) is the fraction of variable (check) nodes having j edges.

Richardson and Urbanke introduced the density evolution technique in [38] for evaluating thresholds of LDPC codes as block lengths approach infinity. The density evolution technique tells us a lower bound of a required SNR for error-free decoding averaged over all possible LDPC codes specified by a degree distribution pair.

For the puncturing problem, we shall set up the problem as follows. Variable nodes of a bipartite graph can be grouped in accordance with their edge degrees. Thus, all coded symbols have the same edge degree in a group denoted G_j for $2 \leq j \leq d_l$. We shall randomly puncture a proportion $\pi_j^{(0)}$ of the symbols in G_j, where $\pi_j^{(0)}$ is determined with an optimization. We define $p^{(0)}$ to be the total puncturing fraction, namely $p^{(0)}$ = (the number of punctured variable nodes)/(the number of variable nodes). We extend the distribution pair, $(\lambda(x), \rho(x))$, to include a puncturing distribution, which is $\big(\lambda(x), \rho(x), \pi^{(0)}(x)\big)$, where $\pi^{(0)}(x) = \pi_2^{(0)}x + \pi_3^{(0)}x^2 + \cdots + \pi_{d_l}^{(0)}x^{d_l-1}$ and $0 \leq \pi_j^{(0)} \leq 1$. The puncturing fraction, $p^{(0)}$ is expressed as

$$p^{(0)} = \frac{\sum_{j=2}^{d_l} \pi_j^{(0)} n_j}{n} = \frac{\sum_{j=2}^{d_l} \pi_j^{(0)} \lambda_j}{\sum_{j=2}^{d_l} \lambda_j/j} = \sum_{j=2}^{d_l} \lambda'_j \pi_j^{(0)}. \tag{4.2}$$

where $n_j = |G_j|$ and $n = \sum_{j=2}^{d_l} |G_j|$. The coding rate of punctured LDPC codes specified by a three-tuple distribution ($\lambda(x)$, $\rho(x)$, $\pi^{(0)}(x)$) is $r(\lambda, \rho, \pi^{(0)}) = r(\lambda, \rho)/(1 - p^{(0)})$. An ensemble of LDPC codes over an AWGN channel and the mixed channel in [16] are special cases expressed by three-tuple distributions, $(\lambda(x), \rho(x), O(x))$ and

$\big(\lambda(x), \rho(x), e^{(0)}I(x)\big)$, respectively, where $O(x) = \sum_{j=2}^{d_l} 0 \cdot x^{j-1}$, $I(x) = \sum_{j=2}^{d_l} x^{j-1}$ and $e^{(0)}$ is a uniform puncturing/erasure probability.

The proposed puncturing scheme is depicted in Fig. 4.2, and the probability density of the received symbol (r_j) corresponding to a coded symbol (c_j) is shown in Fig. 4.3, where a proportion $\pi_j^{(0)}$ of the symbols in G_j are punctured. Our goal is to design puncturing proportions $\pi_j^{(0)}$'s for all j which optimize (minimize) the SNR threshold for a given puncturing fraction $p^{(0)}$. An equivalent optimization, and the one implemented to find $\pi_j^{(0)}$'s, is to fix the SNR threshold and maximize the puncturing fraction threshold $p^{(0)}$.

The basic idea is as follows: assume that a base (unpunctured) code has been designed for rate R. Since the punctured version of that code has rate $R' > R$, its SNR threshold is higher. So rather than fix the puncturing fraction, we fix the target SNR threshold of the punctured code and optimize (maximize) the puncturing fraction $p^{(0)}$ and puncturing proportions $\pi_j^{(0)}$'s for all j for that threshold. As we shall see shortly, the result is a sequence of punctured codes that have thresholds close to those of a set of codes optimally designed for each rate.

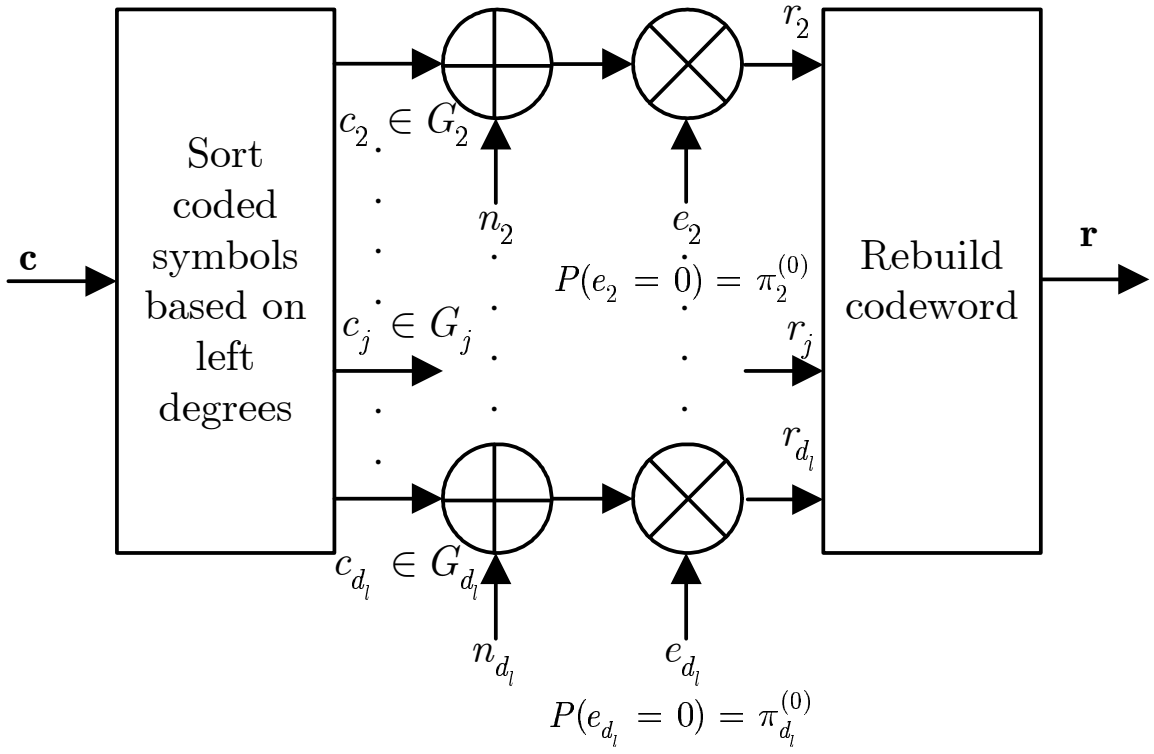


Figure 4.2: Block diagram of the puncturing scheme, $\mathbf{c}$ is a codeword, $c_j \in \{-1, +1\}$ is a coded symbol in G_j, $\mathbf{r}$ is a received signal vector, r_j is the received signal corresponding to c_j, $n_j \sim \mathcal{N}(0, \sigma_n^2)$, $e_j \in \{0, 1\}$, and $P(e_j = 1) = 1 - P(e_j = 0)$.

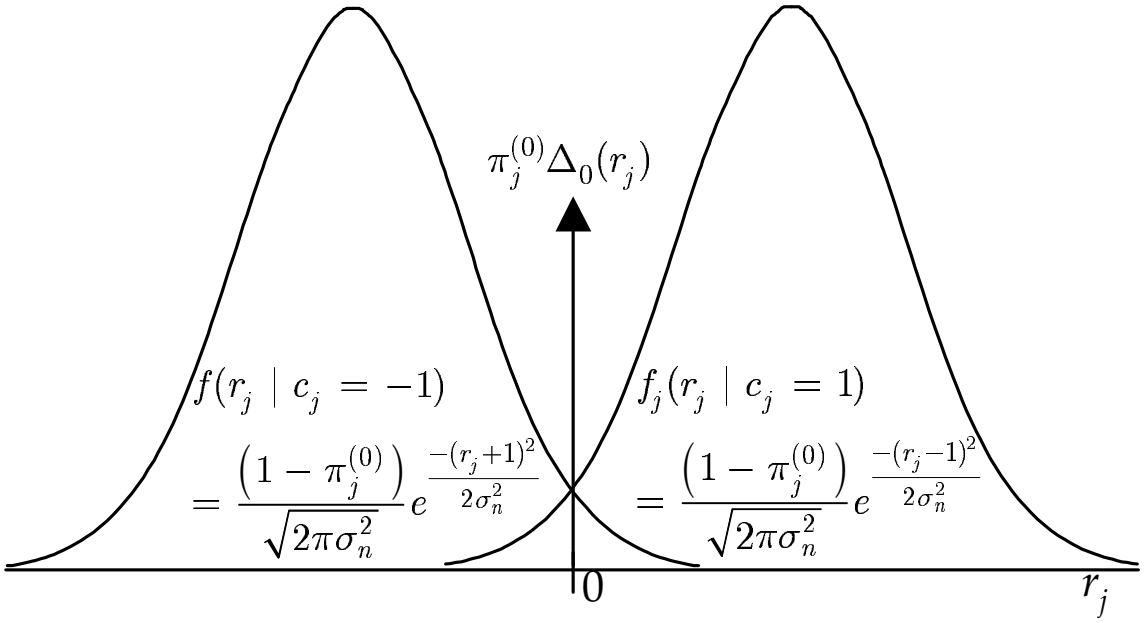


Figure 4.3: Probability density of the received symbol corresponding to c_j, $c_j \in \{-1, +1\}$ is a coded symbol in G_j, $\pi_j^{(0)}$ is a puncturing proportion, r_j is a received symbol, and σ_n is the standard deviation of white Gaussian noise.

For the variable nodes in G_j, the probability density of the log-likelihood ratio (LLR) message can be expressed as

$$\begin{aligned} f(v_j) &= \frac{1-\pi_j^{(0)}}{\sqrt{4\pi m_{u0}}} e^{-\frac{(v_j \pm m_{u0})^2}{4m_{u0}}} + \pi_j^{(0)} \Delta_0(v_j) \\ &= g^{(0)}(v_j) + \pi_j^{(0)} \Delta_0(v_j) \end{aligned}$$

where $v_j = \log_e[p(r_j|c_j = +1)/p(r_j|c_j = -1)]$, $m_{u_0} = E[v_j|v_j \neq 0] = E[2r_j/\sigma_n^2|r_j \neq 0]$, $\sigma_V^2 = \mathrm{Var}(v_j|v_j \neq 0) = 2m_{u_0}$, $\pi_j^{(0)}$ is the random puncturing proportion of coded symbols in G_j, and $\Delta_x(v_j) = \delta(v_j - x)$ is a shifted delta function.

The remainder of this paper is organized into four sections. In Section 4.1, we introduce the channel model we consider and define terminologies for the sequel sections. In Section 4.2, we analyze the thresholds of punctured LDPC codes with Gaussian Approximation (GA) in [8]. The analytic results are useful to understand the convergence of the punctured LDPC codes and can be further simplified with a proper assumption. The simplified recursive equation, called a *steady-state equation*, tells us how the punctured LDPC codes perform and how to design the puncturing distributions. Thus, the analysis gives us not only a prediction method of the thresholds of the punctured LDPC codes but also a design rule of optimal puncturing distributions. In Section 4.3, we design puncturing distributions for two LDPC codes which are designed in [37] and [6]. We also implement the LDPC codes with a code block length of 131072 and apply the designed puncturing distributions to the implemented LDPC codes. Through the simulation, we confirm consistency between the asymptotic and implemented performance. Finally, in Section 4.4, we summarize our work.

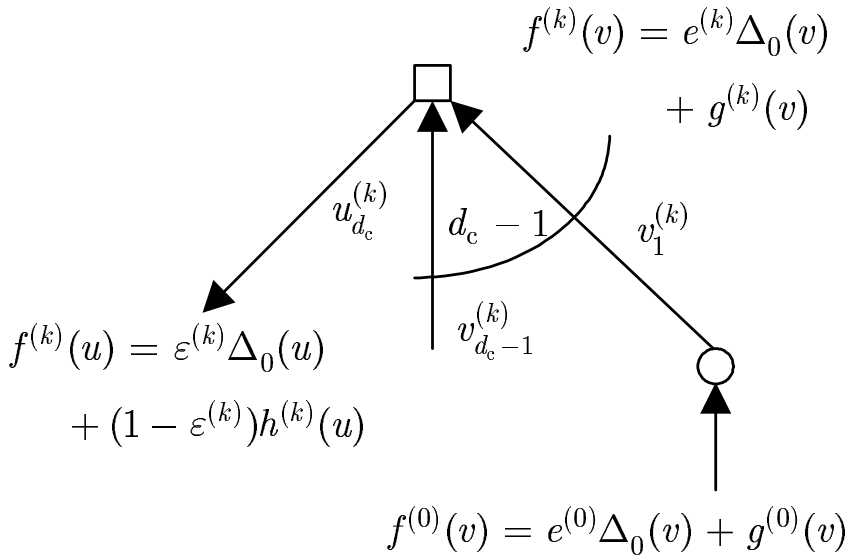
4.2 Puncturing Analysis with Gaussian Approximation

The message flows between a check and a variable nodes are depicted in Fig. 4.4(a) and 4.4(b) where the square and the circle symbols represent check and variable nodes

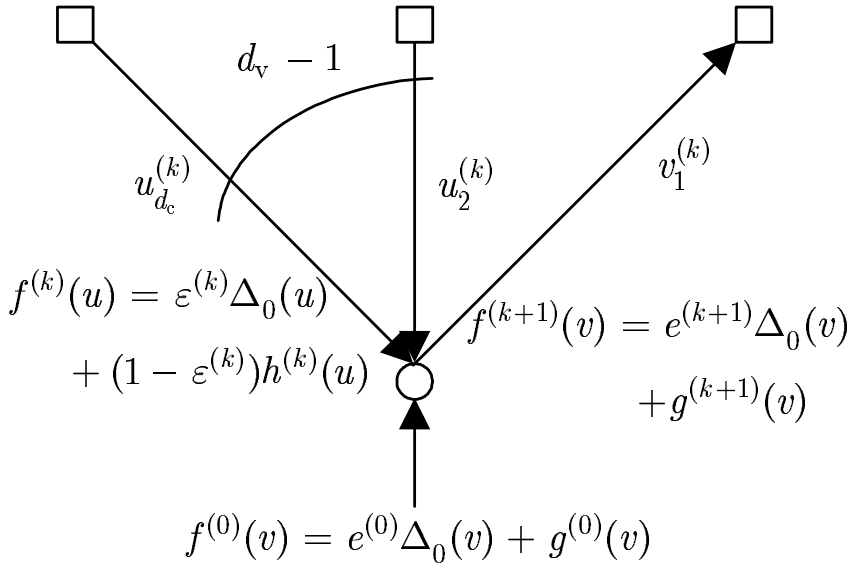
respectively. Assuming the decoder knows the positions of the punctured symbols, the decoder will insert a zero LLR into the message node in place of the channel outputs to initialize the decoder. The function, $g^{(k)}(v)$ $((1-\varepsilon^{(k)})h^{(k)}(u))$ is the continuous part of the probability density of a variable (check) node message. The term, $e^{(k)}$ $(\varepsilon^{(k)})$ is the probability for a variable (check) node LLR message to be equal to zero, d_v (d_c) is the number of neighbors of a variable (check) node, and $v_j^{(k)}$ $(u_j^{(k)})$ is the message of a variable (check) node emitting through the jth edge, respectively (we will drop the edge index j without loss of generality hereafter). In the sum-product decoding algorithm, if at least one of variable nodes connected to a check node is punctured, the check node has zero LLR as its message. Thus, the probability of $\varepsilon^{(k)}$ is expressed as

$$\begin{aligned}
\varepsilon^{(k)} &= \sum_{s=2}^{d_r} P(d_{\mathrm{c}}=s)\varepsilon_{d_{\mathrm{c}}=s}^{(k)} \\
&= \sum_{s=2}^{d_r} \rho_s \left(1-(1-e^{(k)})^{s-1}\right) = 1-\rho\left(1-e^{(k)}\right) \\
\varepsilon_{d_{\mathrm{c}}=s}^{(k)} &= P\left(u^{(k)}=0|d_{\mathrm{c}}=s\right) \\
&= 1-\prod_{i=1}^{s-1} P\left(v^{(k)}\neq 0\right) = 1-\left(1-e^{(k)}\right)^{s-1}
\end{aligned}$$

where $\varepsilon_{d_{\mathrm{c}}=s}^{(k)} = P(u^{(k)}=0|d_{\mathrm{c}}=s)$ is the probability that a check node having s edges sends zero LLR message to a variable node connected to the check node.



(a)



(b)

Figure 4.4: Message flows between a check and a variable nodes. (a) LLR message of a check node at the kth iteration. (b) LLR message of a variable node at the $(k+1)$th iteration.

If none of variable nodes connected to a check node are punctured, the probabilistic characteristics of the check message can be approximated as Gaussian that is denoted as $h^{(k)}(u)$ in Fig. 4.4(a) and 4.4(b). The Gaussian density of LLR message is determined by one variable, the updated mean value $m_u^{(k)}$ at the kth iteration. The updated mean is determined from the following relation:

$$\begin{aligned} m_u^{(k)} &= \sum_{s=2}^{d_c} P\left(d_c = s\right) m_{u|d_c=s}^{(k)} \\ &= \sum_{s=2}^{d_c} \rho_s m_{u|d_c=s}^{(k)}, \text{ and} \\ m_{u|d_c=s}^{(k)} &= \phi^{-1}\Bigg(1- \\ &\qquad E\left[\tanh\left[u^{(k)}/2\right]\left|u^{(k)} \neq 0, d_c = s\right.\right]\Bigg) \\ &= \phi^{-1}\Bigg(1 - E\left[\tanh\left[v^{(k)}/2\right]\left|v^{(k)} \neq 0\right.\right]^{s-1}\Bigg), \end{aligned}$$

where $m_{u|d_c=s}^{(k)}$ is the conditional updated mean given the number of edges of a check node is s.

We can express the updated mean value as

$$\begin{aligned} m_u^{(k)} = \sum_{s=2}^{d_r} \rho_s \phi^{-1} \Bigg(1 - \frac{1}{\left(1-e^{(k)}\right)^{s-1}} \times \\ \left\{\left\langle \tanh\left[v^{(k)}/2\right], g^{(k)}(v)\right\rangle\right\}^{s-1}\Bigg), \text{ and} \end{aligned} \tag{4.3}$$

$$\begin{aligned} &E\left[\tanh\left[v^{(k)}/2\right]\left|v^{(k)} \neq 0\right.\right]^{s-1} \\ &= \frac{1}{\left(1-e^{(k)}\right)^{s-1}} \left\{\left\langle \tanh\left[v^{(k)}/2\right], g^{(k)}(v)\right\rangle\right\}^{s-1}, \end{aligned}$$

where $\langle f(x), g(x)\rangle = \int_{\mathbb{R}} f(x)g(x)\mathrm{d}x$.

To make (4.3) a recursive equation of $m_u^{(k)}$, we need an expression for the probability density of a variable node message in terms of $m_u^{(k-1)}$. In the log-probability domain, the variable node message is determined by the linear sum of $d_v - 1$ incident check node messages as shown in Fig. 4.4. Thus, the probability that the message of

a variable node is zero can be computed as

$$\begin{aligned} e^{(k)} &= P\left(v^{(k)}=0\right) \\ &= \sum_{j=2}^{d_l} P\left(d_{\mathrm{v}}=j\right) e_j^{(k)} \\ &= \sum_{j=2}^{d_l} \lambda_j \pi_j^{(0)} \left(\varepsilon^{(k-1)}\right)^{j-1} \\ &= \lambda^{\pi}\left(\varepsilon^{(k-1)}\right) = \lambda^{\pi}\left(1-\rho\left(1-e^{(k-1)}\right)\right), \end{aligned} \tag{4.4}$$

where $e_j^{(k)}$ is the conditional probability that all incident check node message are zeroes given the variable node has j edges, $e_j^{(0)} = \pi_j^{(0)}$, $\varepsilon^{(-1)} = 1$, $\lambda^{\pi}(x) = \sum_{j=2}^{d_l} \lambda_j^{\pi} x^{j-1}$ and $\lambda_j^{\pi} = \lambda_j \pi_j^{(0)}$.

The residual puncturing proportion ($\pi_j^{(k)}$) and fraction ($p^{(k)}$) at the kth iteration can be computed as

$$\begin{aligned} \pi_j^{(k)} &= \pi_j^{(0)} \left(\varepsilon^{(k-1)}\right)^j, \text{ and} \\ p^{(k)} &= \sum_{j=2}^{d_l} \lambda_j' \pi_j^{(k)}. \end{aligned} \tag{4.5}$$

Thus, the puncturing fraction $p^{(0)}$ in (4.2) is $p^{(k)}$ at $k=0$ with $\varepsilon^{(-1)} = 1$.

The probability density of a variable node message can be factored into three terms as shown in (4.6) The first term gives the probability in (4.4), the second term is made up of the incident check node messages given the variable node is punctured and the last term is the combination of the check node messages with the unpunctured received message. The last two terms comprise the continuous part, used in (4.3).

$$\begin{aligned} f^{(k)}(v) &= e^{(k)}\Delta_0(v) + g^{(k)}(v) \\ &= e^{(k)}\Delta_0(v) \\ &+ \left\{ \sum_{j=2}^{d_l} \lambda_j^{\pi} \sum_{i=1}^{j-1} {}_{(j-1)}\chi_i^{(k)} \mathcal{N}\left(im_u^{(k-1)}, 2im_u^{(k-1)}\right) \right\} \\ &+ \left\{ \sum_{j=2}^{d_l} \lambda_j^{(1-\pi)} \sum_{i=0}^{j-1} {}_{(j-1)}\chi_i^{(k)} \times \right. \\ &\quad \left. \mathcal{N}\left(im_u^{(k-1)} + m_{u_0}, 2im_u^{(k)} + 2m_{u_0}\right) \right\}, \end{aligned} \tag{4.6}$$

where $\lambda_j^{(1-\pi)} = \left(1-\pi_j^{(0)}\right)\lambda_j, \mathcal{N}(m,2m) = \frac{1}{\sqrt{4\pi m}} e^{-\frac{(v-m)^2}{4m}}$, ${}_n\chi_m^k = {}_n\mathrm{C}_m \left(\varepsilon^{(k-1)}\right)^{n-m} \left(1-\varepsilon^{(k-1)}\right)^m$ and ${}_n\mathrm{C}_m$ is the binomial coefficient. The continuous part of a variable node message density derived in (4.6) expand (4.3) to

$$\begin{aligned} m_u^{(k)} = \sum_{s=2}^{d_r} \rho_s \phi^{-1}\Bigg(1 - \frac{1}{\left(1-e^{(k)}\right)^{s-1}} \times \\ \Bigg[1 - \sum_{j=2}^{d_l} \Bigg\{ \lambda_j^{\pi} \sum_{i=0}^{j-1} {}_{(j-1)}\chi_i^{(k)} \phi\left(i m_u^{(k-1)}\right) \\ + \lambda_j^{(1-\pi)} \sum_{i=0}^{j-1} {}_{(j-1)}\chi_i^{(k)} \phi\left(i m_u^{(k-1)} + m_{u_0}\right) \Bigg\} \Bigg]^{s-1} \Bigg), \end{aligned} \tag{4.7}$$

where $\langle \tanh(v/2), \mathcal{N}(x,2x)\rangle = \int_{\mathbb{R}} \tanh(v/2) \frac{1}{\sqrt{4\pi x}} e^{-\frac{(v-x)^2}{4x}} dv = 1-\phi(x)$. The updated mean becomes a recursive equation in (4.7).

Because we model the message densities as Gaussian, the bit error probability can be computed as a weighted sum of Q functions,

$$\begin{aligned} & P_e^{(k)} \\ & = \sum_{j=2}^{d_l} \lambda'_j \pi_j^{(0)} \sum_{i=0}^{j} {}_j\chi_i^{(k)} Q\left(\sqrt{\frac{i m_u^{(k)}}{2}}\right) \\ & + \sum_{j=2}^{d_l} \lambda'_j \left(1-\pi_j^{(0)}\right) \sum_{i=0}^{j} {}_j\chi_i^{(k)} Q\left(\sqrt{\frac{i m_u^{(k)} + m_{u_0}}{2}}\right) \end{aligned} \tag{4.8}$$

$$= \underbrace{\frac{\sum_{j=2}^{d_l} \left(\lambda'_j \pi_j^{(0)} \left(\varepsilon^{(k)}\right)^j\right) = p^{(k)}}{2}}_{\text{Unrecovered punctured symbols}} \qquad (= P_{e_1}^{(k)})$$

$$+ \underbrace{\sum_{j=2}^{d_l} \lambda'_j \pi_j^{(0)} \sum_{i=1}^{j} {}_j\chi_i^{(k)} Q\left(\sqrt{\frac{i m_u^{(k)}}{2}}\right)}_{\text{Recovered punctured symbols}} \qquad (= P_{e_2}^{(k)})$$

$$+ \underbrace{\sum_{j=2}^{d_l} \lambda'_j \left(1-\pi_j^{(0)}\right) \sum_{i=0}^{j} {}_j\chi_i^{(k)} Q\left(\sqrt{\frac{i m_u^{(k)} + m_{u_0}}{2}}\right)}_{\text{Unpunctured symbols}} \quad (= P_{e_3}^{(k)}),$$

where $\sum_{j=2}^{d_l} \lambda'_j \pi_j^{(0)} \chi_0^{(k)} Q(0) = \sum_{j=2}^{d_l} \left(\lambda'_j \pi_j^{(0)} \left(\varepsilon^{(k)} \right)^j \right) \Big/ 2$. In the error probability, the first term describes the half of the unrecovered punctured fraction at the kth iteration, the second term is the error probability due to the punctured symbols and the last term comes from the unpunctured coded symbols.

We may puncture only the parity part of a codeword. Thus, it is enough to recover the coded symbols in the unpunctured part, which is the last term ($P_{e_3}^{(k)}$) in (4.8). However, the following proposition tells that the first two terms also become zero when the last term does.

Proposition 1 *If the unpunctured symbols converge to the correct coded symbols, the punctured symbols also converge to their coded symbols. Simply stated,* $P_{e_3}^{(k)} \to 0 \Leftrightarrow P_e^{(k)} \to 0$.

Proof: Assume that the unpunctured symbols converge to the correct coded symbols, which means the last term in (4.8) converges to zero. Thus,

$$P_{e_1}^{(k)} = \sum_{j=2}^{d_l} \lambda'_j \left(1 - \pi_j^{(0)}\right) \sum_{i=0}^{j} {}_j\chi_i^{(k)} Q\left(\sqrt{\frac{i m_u^{(k)} + m_{u_0}}{2}}\right)$$

$\to 0$ implies

$$\sum_{j=2}^{d_l} \lambda'_j \left(1 - \pi_j^{(0)}\right) \sum_{i=1}^{j} {}_j\chi_i^{(k)} Q\left(\sqrt{\frac{i m_u^{(k)} + m_{u_0}}{2}}\right)$$

$\to 0$ and

$$\sum_{j=2}^{d_l} \lambda'_j \left(1 - \pi_j^{(0)}\right) \left(\varepsilon^{(k-1)}\right)^{j-1} Q\left(\sqrt{\frac{m_{u_0}}{2}}\right) \to 0,$$

when $m_u^{(k)} \to \infty$ and $\varepsilon^{(k-1)} \to 0$, respectively. The recovered punctured symbols converge to correct coded symbols because

$$P_{e_2}^{(k)} = \sum_{i=1}^{j} {}_j\chi_i^{(k)} Q\left(\sqrt{\frac{i m_u^{(k)}}{2}}\right) \to 0,$$

as $m_u^{(k)} \to \infty$. The fraction of the unrecovered punctured symbols ($P_{e_3}^{(k)}$) also tends to zero due to $\varepsilon^{(k-1)} \to 0$ as $k \to \infty$.

■

The evolution of the puncturing fraction in (4.5) indicates that the residual puncturing fraction during iterations does not depend on SNR but only on a degree distribution pair $(\lambda(x), \rho(x))$ and a puncturing distribution $\pi^{(0)}(x)$, or in short, a three-tuple distribution $\big(\lambda(x), \rho(x), \pi^{(0)}(x)\big)$.

The following proposition gives a sufficient condition for the residual puncturing fraction to converge to zero.

Proposition 2 *If $e^{(k)} \to 0$ (equivalently $p^{(k)}, \to 0$) as $k \to \infty$ for a puncturing distribution $\pi^{(0)}(x)$, then $e^{(k)} \to 0$ for any puncturing distribution $\omega^{(0)}(x)$ such that $\omega_j^{(0)} \le \pi_j^{(0)}$.*

Proof: We will prove this proposition by induction. $e^{(0)}(\omega) \le e^{(0)}(\pi)$ because

$$\begin{aligned} e^{(0)}(\omega) &= \sum_{j=2}^{d_l} \omega_j^{(0)} \lambda_j, \\ e^{(0)}(\pi) &= \sum_{j=2}^{d_l} \pi_j^{(0)} \lambda_j, \text{ and} \\ \omega_j^{(0)} \lambda_j &\le \pi_j^{(0)} \lambda_j. \end{aligned}$$

Suppose $e^{(k)}(\omega) \le e^{(k)}(\pi)$ for $k > 0$, then $e^{(k+1)}(\omega) \le e^{(k+1)}(\pi)$ because

$$\begin{aligned} & e^{(k+1)}(\omega) \\ &= \sum_{j=2}^{d_l} \omega_j^{(0)} \lambda_j \left(1 - \rho\left(1 - e^{(k)}(\omega)\right)\right) \quad \text{(a)} \\ &\le \sum_{j=2}^{d_l} \omega_j^{(0)} \lambda_j \left(1 - \rho\left(1 - e^{(k)}(\pi)\right)\right) \quad \text{(b)} \\ &\le \sum_{j=2}^{d_l} \pi_j^{(0)} \lambda_j \left(1 - \rho\left(1 - e^{(k)}(\pi)\right)\right) \quad \text{(c)} \\ &= e^{(k+1)}(\pi) \end{aligned}$$

The relation between (a) and (b) holds because $0 < e^{(k)}(\omega) \leq e^{(k)}(\pi)$. The inequality between (b) and (c) holds due to $\omega_j^{(0)} \leq \pi_j^{(0)}$.

■

The proposition tells that even if a puncturing distribution has a smaller puncturing fraction, this does not guarantee the convergence of the residual puncturing fraction to zero. Next, we will show a simple example.

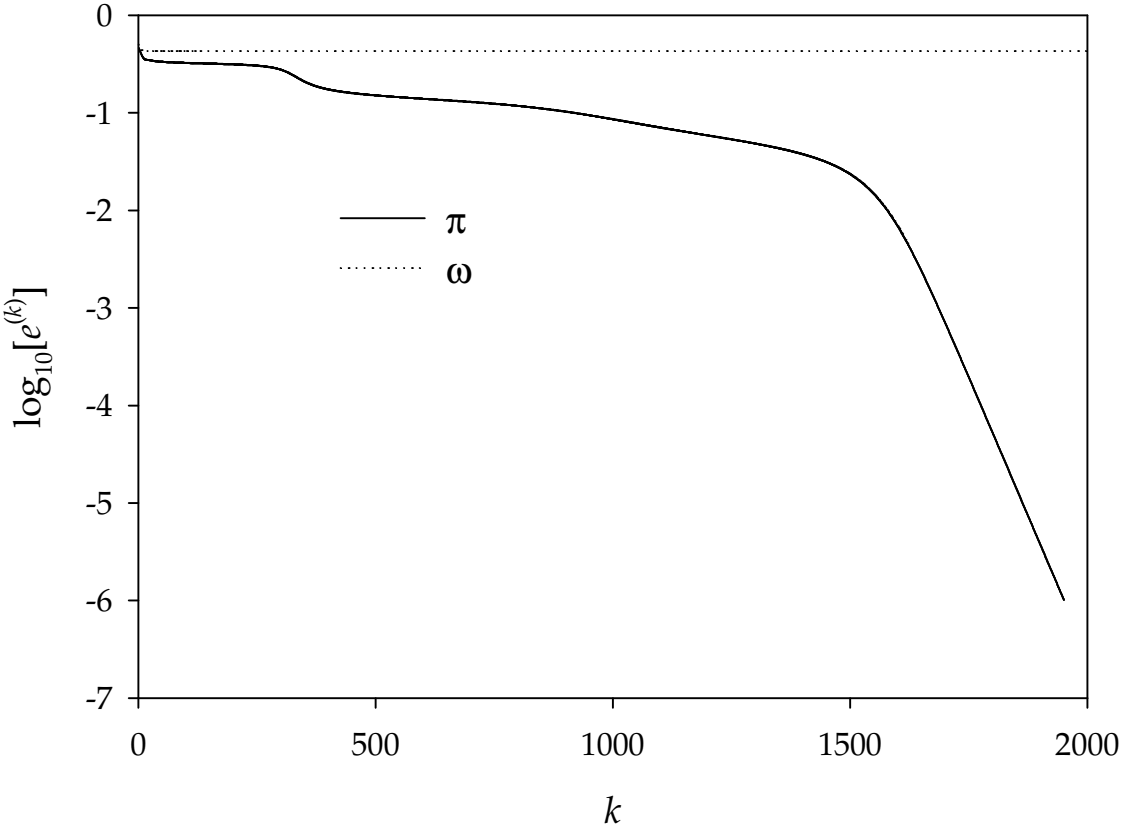


Figure 4.5: $e^{(k)}$ with two puncturing distributions, $\pi^{(0)}(x) = 0.57164x + 0.38346x^2 + 0.75360x^5 + 0.02071x^6 + 0.44034x^{19}$ (denoted as π and $p^{(0)}(\pi) = 0.49336$) and $\omega^{(0)}(x) = 0.46000x + 0.38346x^2 + 0.75360x^5 + 0.02071x^6 + 0.65000x^{19}$ (denoted as ω and $p^{(0)}(\omega) = 0.45257$) for a degree distribution pair $\lambda(x) = 0.23403x + 0.21242x^2 + 0.14690x^5 + 0.10284x^6 + 0.30381x^{19}$ and $\rho(x) = 0.71875x^7 + 0.28125x^8$.

Example 1 *For a degree distribution pair,* $\lambda(x) = 0.23403x + 0.21242x^2 + 0.14690x^5 + 0.10284x^6 + 0.30381x^{19}$, *and* $\rho(x) = 0.71875x^7 + 0.28125x^8$, *puncturing fractions* $p^{(0)}(\pi) = 0.49336$ *and* $p^{(0)}(\omega) = 0.45257$ *can be made of puncturing distributions* $\pi^{(0)}(x) = 0.57164x + 0.38346x^2 + 0.75360x^5 + 0.02071x^6 + 0.44034x^{19}$ *and* $\omega^{(0)}(x) = 0.46000x + 0.38346x^2 + 0.75360x^5 + 0.02071x^6 + 0.65000x^{19}$, *respectively. Although* $p^{(0)}(\omega) < p^{(0)}(\pi)$, $e^{(k)}(\pi) \to 0$ *but* $\exists\, \epsilon > 0$ *such that* $e^{(k)}(\pi) > \epsilon$, $\forall k > 0$. *See Fig. 4.5.*

If $e^{(k)}$ ($\varepsilon^{(k)}$) converges to zero, we can approximate ${}_j\chi_i$ as δ_{ji}, which simplifies the recursive equation (4.7) as

$$m_u^{(k)} = \sum_{s=2}^{d_r} \rho_s \phi^{-1}\left(1 - \left[1 - \sum_{j=2}^{d_l} \left\{\lambda_j^{\pi}\phi\left((j-1)m_u^{(k-1)}\right) + \lambda_j^{(1-\pi)}\phi\left((j-1)m_u^{(k-1)} + m_{u_0}\right)\right\}\right]^{s-1}\right). \tag{4.9}$$

We call this simplified equation a *steady-state equation.*

For error-free decoding ($P_e^{(k)} \to 0$ in (4.8)), the recursive equation must grow to infinity, which are $m_u^{(k)} < m_u^{(k+1)}$ for any $k \geq 0$ and $m_u^{(k)} \to \infty$ as $k \to \infty$. This inequality ($m_u^{(k)} < m_u^{(k+1)}$) can be equivalently expressed as

$$\begin{aligned} r &> \sum_{j=2}^{d_l} \lambda_j^{\pi} h_j\left(0, r\right) + \lambda_j^{(1-\pi)} h_j\left(m_{u_0}, r\right) \\ &= h\left(0, \lambda^{\pi}(x), r\right) + h\left(m_{u_0}, \lambda^{(1-\pi)}(x), r\right) \\ &= H\left(m_{u_0}, \lambda^{\pi}(x), r\right), \forall r \in (0, \phi\left(m_{u_0}\right)], \end{aligned} \tag{4.10}$$

where $h_j(s, r)$ is defined in Section 2.4, $h\left(s, \lambda^q(x), r\right) = \sum_{j=2}^{d_l} \lambda_j^q h_j(s, r) = E_{\lambda_j^q}\left[h_j(s, r)\right]$, q is either π or $1 - \pi$ and $H\left(s, \lambda^{\pi}(x), r\right) = h\left(0, \lambda^{\pi}(x), r\right) + h\left(s, \lambda^{1-\pi}(x), r\right)$.

The inequality in (4.10) give us an insight into a design rule for good puncturing distributions. We explain the steady-state equation with a graphical interpretation

by evaluating a 1/2 rate irregular code whose degree distribution pair is

$$\lambda(x) = 0.25105x + 0.30938x^2 + 0.00104x^3 + 0.43853x^9,$$

and

$$\rho(x) = 0.63676x^6 + 0.36324x^7,$$

which was designed using the density evolution technique in [37].

In Fig. 4.6, the solid and the long-dashed lines represent $\{h_j(s,r) - r\}$ and $\{h_j(0,r) - r\}$ for $j = 2, \ldots, 10$ from the top to the bottom and $s = 2.76700$, respectively. By averaging $h_j(s,r)$ and $h_j(0,r)$ with $\lambda_j^{(1-\pi)}$ and λ_j^{π}, we can compute $h(s, \lambda^{(1-\pi)}(x), r)$ and $h(s, \lambda^{\pi}(x), r)$, respectively. Finally, $H(s, \lambda^{\pi}(x), r)$ is calculated by summing $h(s, \lambda^{(1-\pi)}(x), r)$ and $h(s, \lambda^{\pi}(x), r)$. To satisfy the inequality in (4.10), $\{H(s, \lambda^{\pi}(x), r) - r\}$ must be less than zero when $0 < r \leq \phi(s)$. In summing $h_j(s,r)$ and $h_j(0,r)$, we have freedom to weight $h_j(s,r)$ and $h_j(0,r)$ by optimizing a puncturing proportion, $\pi_j^{(0)}$ for a given m_{u_0}, which is equivalent to choosing λ_j^{π} and $\lambda_j^{(1-\pi)}$ to minimize m_{u_0} for a given $p^{(0)}$. In Fig. 4.7, we can see that $\{H(s, \lambda^{\pi}(x), r) - r\}$ barely averts touching zero, which is a fixed point of the recursive equation (in this example, $\min |H(s, \lambda^{\pi}(x), r) - r| = 10^{-4}$). The smaller gap between $\{H(s, \lambda^{\pi}(x), r) - r\}$ and the r axis makes a decoder iterate more for achieving error-free decoding, although the SNR threshold may be able to decrease.

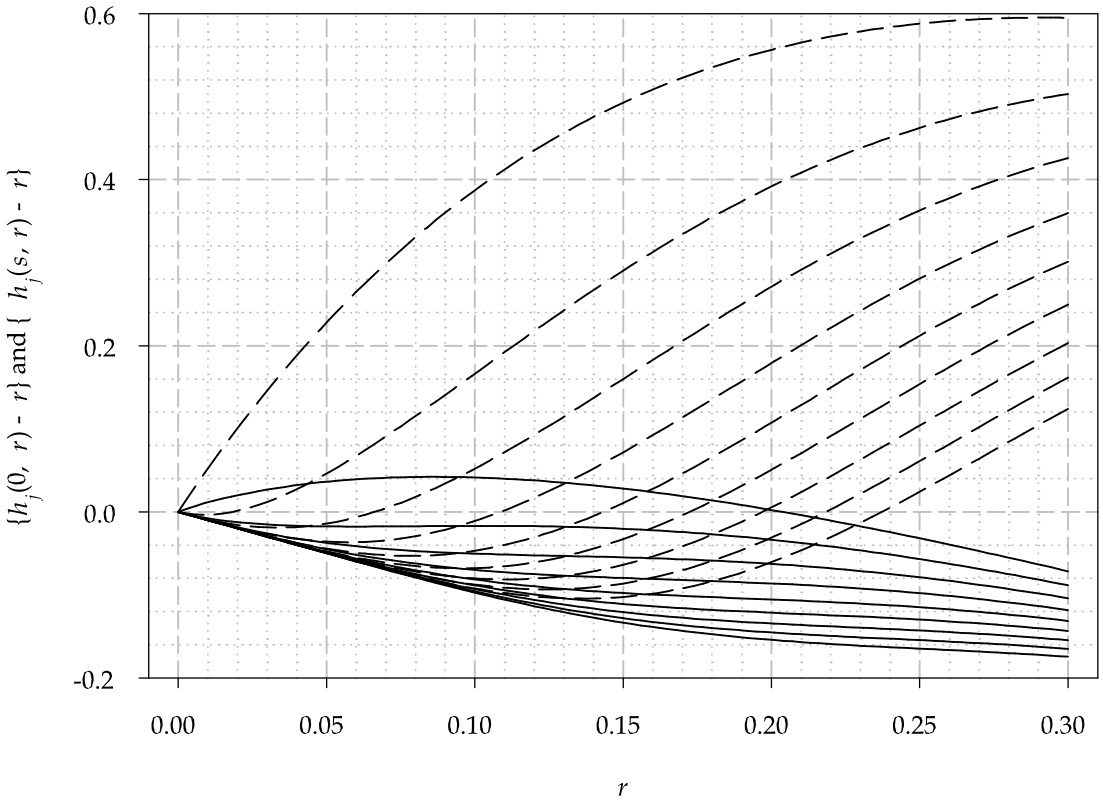


Figure 4.6: $\{h_j(s,r) - r\}$ for $j = 2, \ldots, 10$ (top to bottom), $s = 0$ (long-dash) and 2.76700 (solid line).

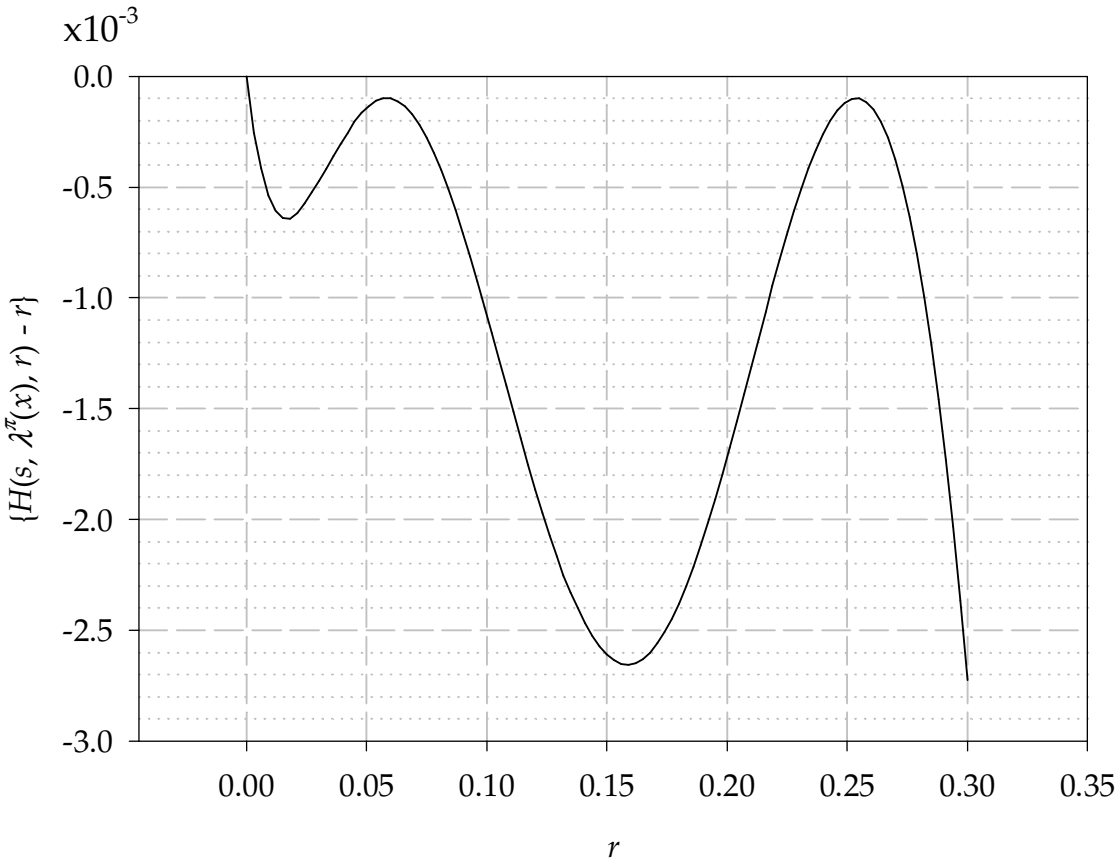


Figure 4.7: $\{H(s, \lambda^{\pi}(x), r) - r\}$ and $\min |H(s, \lambda^{\pi}(x), r) - r| = 10^{-4}$ for $s = 2.76700$ and $p^{(0)} = 0.25$.

We can rewrite (4.10) as

$$\underbrace{\sum_{j=2}^{d_l} \lambda_j \pi_j^{(0)} \left(h_j\left(0, r\right) - h_j\left(m_{u_0}, r\right) \right)}_{\text{a linear combination of } \pi_j^{(0)}\text{'s}} + \underbrace{\sum_{j=2}^{d_l} \lambda_j h_j\left(m_{u_0}, r\right)}_{\text{a constant}} < r,$$

where the inequality becomes a linear combination of $\pi_j^{(0)}$'s. Thus, we can design a puncturing distribution for a given degree distribution pair with linear programming, which is formulated in Fig. 4.10.

To use the design rule, the designed puncturing distribution must satisfy the required condition for the steady-state equation, $e^{(k)} \to 0$ as $k \to \infty$. If we design $\pi^{(0)}(x)$ with the design rule in Fig. 4.10, we can say that $e^{(k)} \to 0$ as $k \to \infty$ in most practical cases. We shall prove the claim in a heuristic way by showing that a designed puncturing distribution $\pi^{(0)}(x)$ satisfies the following inequality;

$$\begin{aligned} & \sum_{j=2}^{d_l} \lambda_j \pi_j^{(0)} \left(1 - \sum_{i=2}^{d_r} \rho_i (1-r)^{i-1} \right)^{j-1} \\ = & \sum_{j=2}^{d_l} \lambda_j \pi_j^{(0)} e_j(r) \\ \leq & \sum_{j=2}^{d_l} \left\{ \lambda_j \pi_j^{(0)} h_j\left(0, r\right) + \lambda_j (1 - \pi_j^{(0)}) h_j\left(m_{u_0}, r\right) \right\} < r, \end{aligned}$$

where $e_j(r) = (1 - \sum_i \rho_i (1-r)^{i-1})^{j-1}$. A sufficient condition for the inequality is $e_j(r) \leq h_j(0, r)$ because

$$\begin{aligned} & \sum_{j=2}^{d_l} \lambda_j \pi_j^{(0)} h_j\left(0, r\right) \\ \leq & \sum_{j=2}^{d_l} \left\{ \lambda_j \pi_j^{(0)} h_j\left(0, r\right) + \lambda_j (1 - \pi_j^{(0)}) h_j\left(m_{u_0}, r\right) \right\}, \end{aligned}$$

where $0 \leq h_j\left(m_{u_0}, r\right) \leq 1$ and $0 \leq \pi_j^{(0)} \leq 1$. In most practical cases, right degree distributions for AWGN channels have the form $\rho(x) = \rho x^{d_r - 2} + (1 - \rho) x^{d_r - 1}$ [7], and d_l is large enough for the approximation

$$\begin{aligned} & \sum_{i=2}^{d_r} \rho_i \phi^{-1}\left(1-(1-r)^{i-1}\right) \\ \approx\ & \phi^{-1}\left(\sum_{i=2}^{d_r} \rho_i \left(1-(1-r)^{i-1}\right)\right) \\ =\ & \phi^{-1}\left(1-\sum_{i=2}^{d_r} \rho_i (1-r)^{i-1}\right), \end{aligned} \tag{4.11}$$

because $(1-r)^{d_r-2} - (1-r)^{d_r-1} = r(1-r)^{d_l-2} \ll 1$ and $\phi^{-1}(\cdot)$ can be linearized between $1-(1-r)^{d_r-2}$ and $1-(1-r)^{d_r-1}$. Thus, we can approximate $h_j(0,r)$ as

$$\begin{aligned} h_j(0,r) &= \phi\left((j-1)\sum_{i=2}^{d_r} \rho_i \phi^{-1}\left(1-(1-r)^{i-1}\right)\right) \\ &\approx \phi\left((j-1)\phi^{-1}\left(1-\sum_{i=2}^{d_r} \rho_i (1-r)^{i-1}\right)\right), \end{aligned}$$

and rewrite $e_j(r)$ as

$$e_j(r) = \phi\left(\phi^{-1}\left(\left[1-\sum_{i=2}^{d_r} \rho_i (1-r)^{i-1}\right]^{j-1}\right)\right).$$

If $x^n = \phi\left(\phi^{-1}(x^n)\right) \leq \phi\left(n\phi^{-1}(x)\right)$ for $0 \leq x \leq 1$ and $n \geq 1$, we can say that $e_j(r) \leq h_j(0,r) \leq h_j(0,r) + h_j(m,r)$. In the following theorem we show that the inequality, $\phi\left(\phi^{-1}(x^n)\right) \leq \phi\left(n\phi^{-1}(x)\right)$ holds true by using the convexity of $\phi(x)$.

Theorem 1 *$x^n = \phi\left(\phi^{-1}(x^n)\right) \leq \phi\left(n\phi^{-1}(x)\right)$ for $0 \leq x \leq 1$ and $n \geq 1$.*

Proof: We will prove this theorem by induction. The equality holds at $x = 1$ and 0 for any $n \geq 1$ because $\phi^{-1}(1) = 0$, $\phi(0) = 1$, $\phi^{-1}(0) = \infty$ and $\phi(\infty) = 0$. For $n = 1$, $x = \phi\left(\phi^{-1}(x)\right)$. Suppose $x^n \leq \phi\left(n\phi^{-1}(x)\right)$ for $n > 1$, then $x^{n+1} = x \cdot x^n \leq x \cdot \phi\left(n\phi^{-1}(x)\right)$. Because $\phi(x)$ is a continuous convex function of x, we can have the following inequality [1];

$$\frac{\phi(x)-\phi(w)}{x-w} \leq \frac{\phi(z)-\phi(y)}{z-y},$$

for $0 \le w < x \le y < z \le 1$. By replacing w, x, y, and z with 0, $\phi^{-1}(x)$, $n\phi^{-1}(x)$, and $(n+1)\phi^{-1}(x)$, respectively, we can rewrite the inequality as

$$\begin{aligned}
& \frac{\phi\left(\phi^{-1}(x)\right)-\phi(0)}{\phi^{-1}(x)-0} \\
= & \frac{\phi\left(\phi^{-1}(x)\right)-1}{\phi^{-1}(x)} \qquad \text{(a)} \\
\le & \frac{\phi\left((n+1)\phi^{-1}(x)\right)-\phi\left(n\phi^{-1}(x)\right)}{(n+1)\phi^{-1}(x)-n\phi^{-1}(x)} \\
= & \frac{\phi\left((n+1)\phi^{-1}(x)\right)-\phi\left(n\phi^{-1}(x)\right)}{\phi^{-1}(x)} \\
\le & \frac{\phi\left((n+1)\phi^{-1}(x)\right)-\phi\left(n\phi^{-1}(x)\right)}{\phi^{-1}(x)\phi\left(n\phi^{-1}(x)\right)} \qquad \text{(b)} \\
= & \frac{\phi\left((n+1)\phi^{-1}(x)\right)/\phi\left(n\phi^{-1}(x)\right)-1}{\phi^{-1}(x)} \qquad \text{(c)}
\end{aligned}$$

where $x \in [0,1)$, $\phi(0)=1$ in (a), and $0 < \phi(a) \le 1$ for any $a \ge 0$ in (b). The relation between (a) and (c) gives us

$$x^{n+1} \le x\phi(n\phi^{-1}(x)) \le \phi\left((n+1)\phi^{-1}(x)\right),$$

for any $n \ge 1$ and $x \in [0,1]$.

■

In Fig. 4.8, we depict $\phi\left(n\phi^{-1}(x)\right)-x^n$, which is always non-negative. We also compare $h_j(0,r)$ (dashed lines) with $e_j(r)$ (solid lines) for a degree distribution pair $\lambda(x)=0.25105x+0.30938x^2+0.00104x^3+0.43853x^9$ and $\rho(x)=0.63676x^6+0.36324x^7$ in Fig. 4.9 where $e_2(r)$ and $h_2(0,r)$ are completely overlapped and $e_j(r) \le h_j(0,r)$ for $j > 2$.

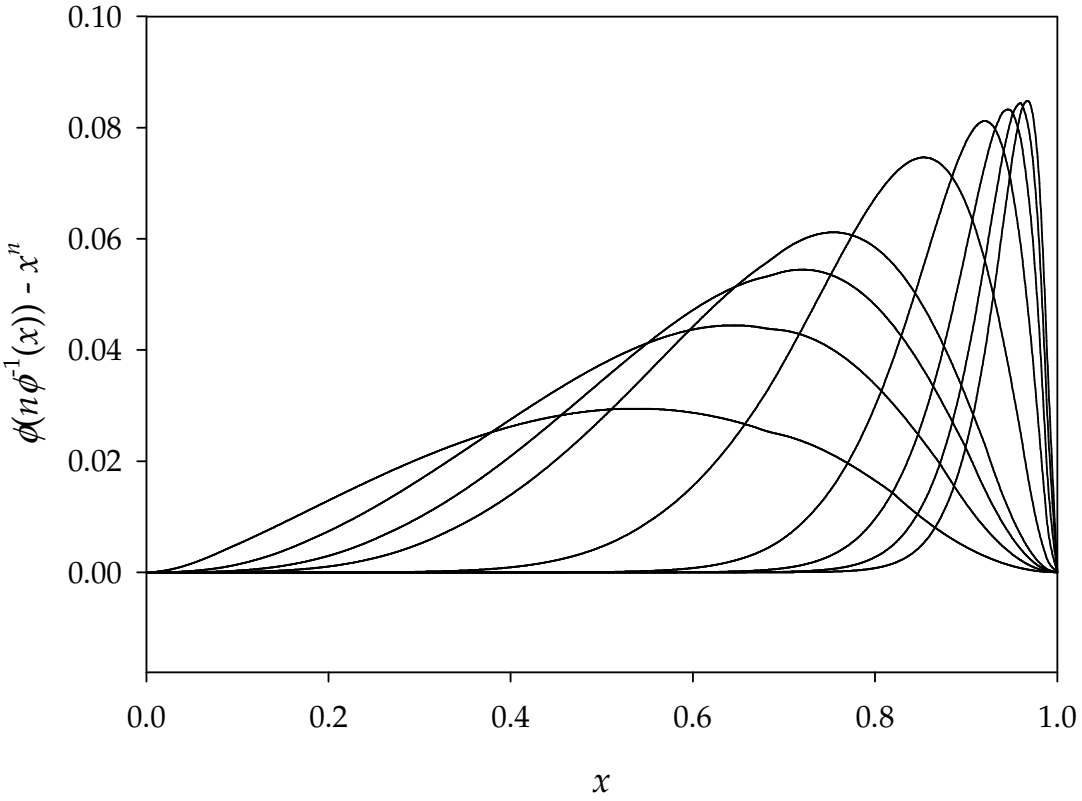


Figure 4.8: $\phi(n\phi^{-1}(x)) - x^n$ for $n = 2$, 3, 4, 5, 10, 20, 30, 40 and 50 from left to right.

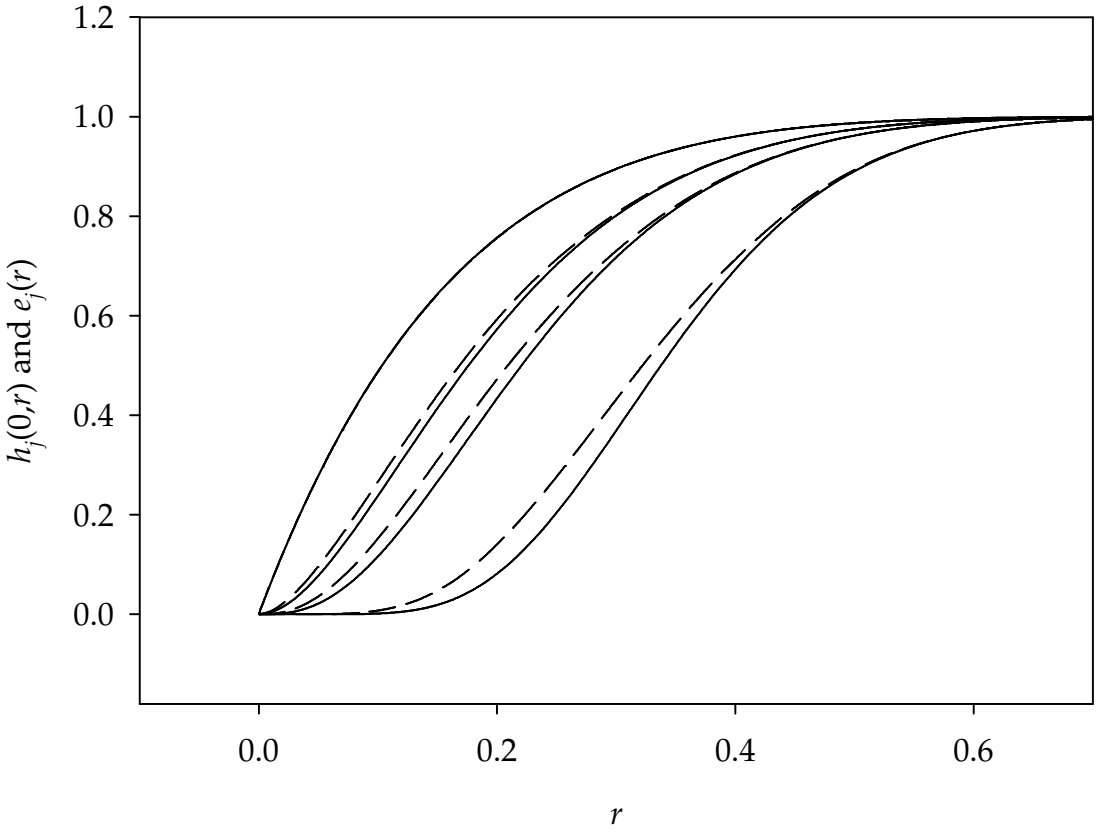


Figure 4.9: $h_j(0, r)$ (dashed lines) and $e_j(r)$ (solid lines) for a degree distribution pair, $\lambda(x) = 0.25105x+0.30938x^2+0.00104x^3+0.43853x^9$ and $\rho(x) = 0.63676x^6+0.36324x^7$ (j = 2, 3, 4, and 10 from left to right).

4.3 Design of Puncturing Distributions for Base $R = 1/2$ Code

In this section, we design puncturing distributions with two good edge degree distribution pairs in [37] and [6] for 1/2-rate codes that have the maximum variable degrees of 10 and 20, respectively. The design goal is to maximize the puncturing fraction, $p^{(0)}$ for a given E_b/N_0.

We optimize puncturing fractions, $p^{(0)}$ from 0.05 to 0.45 in increments of 0.05 to achieve the coding rate, r, from 0.5263 to 0.9091 with the degree distribution pair in [37],

$$\lambda(x) = 0.25105x + 0.30938x^2 + 0.00104x^3 + 0.43853x^9,$$

and

$$\rho(x) = 0.63676x^6 + 0.36324x^7.$$

$$\max_{\pi_j^{(0)}\text{'s}} p^{(0)} = \max_{\pi_j^{(0)}\text{'s}} \frac{\sum_{j=2}^{d_l} \pi_j^{(0)} \lambda_j / j}{\sum_{j=2}^{d_l} \lambda_j / j} = \max_{\pi_j^{(0)}\text{'s}} \left[\bar{n}^T \cdot \bar{\pi}\right]$$

Constraints:

1. $\mathbf{A}\bar{\pi} < \bar{b}$,
2. $\mathbf{0} \leq \bar{\pi} \leq \mathbf{1}$,
3. $m_{u_0} = m^*$

where $\bar{n} = \dfrac{1}{\sum_{j=2}^{d_l} \lambda_j / j} \left[\dfrac{\lambda_2}{2}, \dfrac{\lambda_3}{3}, \ldots, \dfrac{\lambda_{d_l-1}}{d_{l-1}}, \dfrac{\lambda_{d_l}}{d_l}\right]^T$, $\bar{\pi} = \left[\pi_2^{(0)}, \pi_3^{(0)}, \ldots, \pi_{d_l-1}^{(0)}, \pi_{d_l}^{(0)}\right]^T$,

$\mathbf{A} = \left[a_{i,j}\right]$, $a_{i,j} = \lambda_j \left\{h_j(0, r_i) - h_j(m_{u_0}, r_i)\right\}$, for $1 \leq i \leq N$ and $1 \leq j \leq d_l$,

$\bar{b} = \left[r_1 - H(m_{u_0}, r_1), r_2 - H(m_{u_0}, r_2), \ldots, r_{N-1} - H(m_{u_0}, r_{N-1}), r_N - H(m_{u_0}, r_N)\right]^T$,

$r_i = i\dfrac{r_{\max}}{N}$, $H(m_{u_0}, r_i) = \sum_{j=2}^{d_l} \lambda_j h(m_{u_0}, r_i)$, N is a large positive integer and m^* is the maximum allowable signal power.

Figure 4.10: Proposed design rule with linear programming.

In Fig. 4.11, we depict the puncturing proportions, $\pi_j^{(0)}$'s, all but one of which have smooth changes with respect to the rate changes, indicating that the codes can be punctured in a rate-compatible fashion. The abrupt changes of $\pi_4^{(0)}$ result from the small magnitude of λ_4, which means the puncturing of coded symbols in G_4 does not have much influence on the overall performance. The puncturing proportions obtained from linear programming are listed in Table 4.1. To verify the analytic results and the linear programming algorithm, we also designed the puncturing distributions with the Discretized Density Evolution (DDE) [6] technique and Differential Evolution (DE) [36] technique that searches an optimum vector with a combination of a genetic and a hill climbing algorithms. The results are listed in Table 4.2. The DDE algorithm models the message-passing decoder more accurately, although the algorithm lacks analytic insight and needs more computation. We compare the results from linear programming (denoted as LP) with ones from DE in Fig. 4.11 through 4.15. The optimizations with DE proceed in three different ways to avoid possible local minima. In the first trial (denoted as TR1), we search the puncturing distribution over the $\mathbb{U}^4$ space without initial values where $\mathbb{U} = [0, 1]$. In the second and third trials (denoted as TR2 and TR3), we use the puncturing proportions, $\pi_j^{(0)}$'s obtained from LP as initial values that are perturbed with Gaussian and uniform noise, respectively. The comparisons show that there is no difference between the results from the three different trials with DE. However, we can see some discrepancies between the results from LP and DE, although they have the same trends with respect to the coding rates. The discrepancies are more distinctive at higher degrees, which comes from the fact that node distribution is inversely proportional to the degree in (4.1), ($\lambda'_j = \alpha\lambda_j/j$ and $\alpha > 0$). Thus, the lowest degree puncturing proportion, $\pi_2^{(0)}$ is the most important which has a good match with the results from DE in Fig. 4.12.

To see the code performance of the intentionally and randomly punctured LDPC codes, we compute the gaps between the asymptotic E_b/N_0 performances and the

capacity of the BPSK signal with respect to the coding rates in Fig. 4.16. The asymptotic E_b/N_0 performances are evaluated using DDE. We can notice that there is less than 0.1dB difference between the performances of the punctured LDPC code with the puncturing distributions from LP and DE. Although the performance difference between the LDPC code with puncturing distributions from LP and DE is relatively small, we are more interested in the ultimate performance of the punctured LDPC codes. Thus, we describe the performances of the punctured LDPC codes with the puncturing distributions from DE. The results show that the base code has a 0.2dB gap from the capacity and an additional 0.1dB gap permits us to vary the coding rate up to 0.72. The changes in coding rate up to 0.84 and 0.92 are obtained at the expenses of 0.2dB and 0.4dB additional gaps, respectively.

To verify the design rule, we implement the LDPC code and evaluate required E_b/N_0's for a BER of 10^{-4} of the intentional and random puncturing schemes. The implemented code has a code block length of 131072, a coding rate of 0.5 and no 4 cycle loops in the parity check matrix. The simulated E_b/N_0's are evaluated by watching more than 50 error frames. We measure E_b/N_0 gaps of the simulation results as we did in Fig. 4.16 for the asymptotic E_b/N_0 performances of the LDPC code. The gaps of the simulation results and theoretical predictions are compared in Fig. 4.17 which shows a consistency between them. We also plot BER curve's of the intentionally and randomly punctured LDPC codes in Figs. 4.18 and 4.19, respectively.

Table 4.1: Puncturing proportions, $\pi_j^{(0)}$'s for $\lambda(x) = 0.25105x + 0.30938x^2 + 0.00104x^3 + 0.43853x^9$ and $\rho(x) = 0.63676x^6 + 0.36324x^7$ at the overall puncturing probabilities of 0.00, 0.05, 0.10, 0.15, 0.20, 0.25, 0.30, 0.35, 0.40 and 0.45 with linear programming.

σ	0.95574	0.91582	0.87463	0.83283	0.78866	0.74095	0.68875	0.63009	0.56086	0.46754
$\pi_2^{(0)}$	0.00000	0.06944	0.14806	0.20729	0.26188	0.31805	0.37500	0.43221	0.48985	0.55105
$\pi_3^{(0)}$	0.00000	0.00004	0.00021	0.04350	0.09652	0.14650	0.19395	0.24140	0.28801	0.32952
$\pi_4^{(0)}$	0.00000	0.07155	0.05424	0.00373	0.32006	0.06447	0.24312	0.02393	0.01062	0.01762
$\pi_{10}^{(0)}$	0.00000	0.11172	0.19739	0.23732	0.26550	0.29973	0.33503	0.37200	0.40846	0.44665
$p^{(0)}$	0.00000	0.05000	0.10000	0.15000	0.20000	0.25000	0.30000	0.35000	0.40000	0.45000

Table 4.2: Puncturing proportions, $\pi_j^{(0)}$'s for $\lambda(x) = 0.25105x + 0.30938x^2 + 0.00104x^3 + 0.43853x^9$ and $\rho(x) = 0.63676x^6 + 0.36324x^7$ at σ's used in linear programming.

σ	0.95574	0.91582	0.87463	0.83283	0.78866	0.74095	0.68875	0.63009	0.56086	0.46754
$\pi_2^{(0)}$	0.00000	0.07886	0.14628	0.20276	0.25381	0.31767	0.36624	0.41838	0.47074	0.52325
$\pi_3^{(0)}$	0.00000	0.01405	0.04641	0.09305	0.15000	0.18079	0.24119	0.29462	0.34447	0.39074
$\pi_4^{(0)}$	0.00000	0.06081	0.01616	0.03356	0.34406	0.05752	0.49649	0.05265	0.02227	0.01324
$\pi_{10}^{(0)}$	0.00000	0.07206	0.19149	0.16504	0.19149	0.24692	0.27318	0.30975	0.34997	0.39436
$p^{(0)}$	0.00000	0.05324	0.10520	0.15505	0.20462	0.25430	0.30412	0.35378	0.40316	0.45194

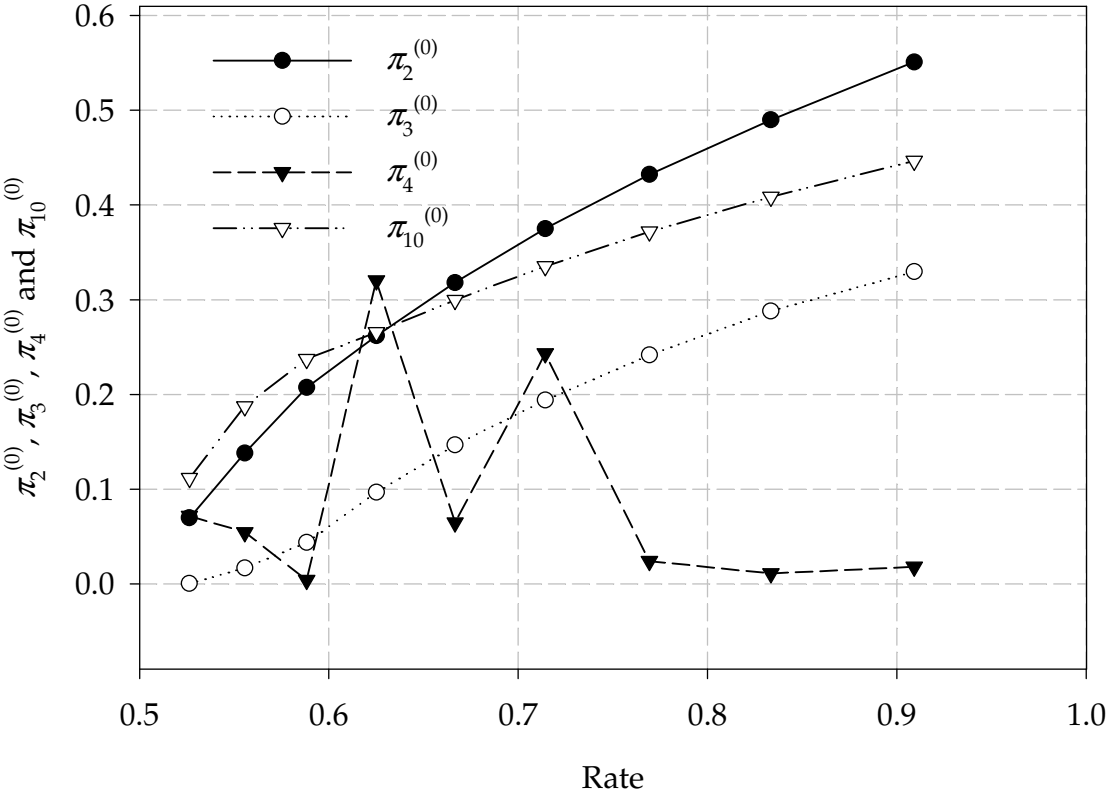


Figure 4.11: Puncturing proportions, $\pi_j^{(0)}$'s with linear programming for a degree distribution pair, $\lambda(x) = 0.25105x + 0.30938x^2 + 0.00104x^3 + 0.43853x^9$ and $\rho(x) = 0.63676x^6 + 0.36324x^7$.

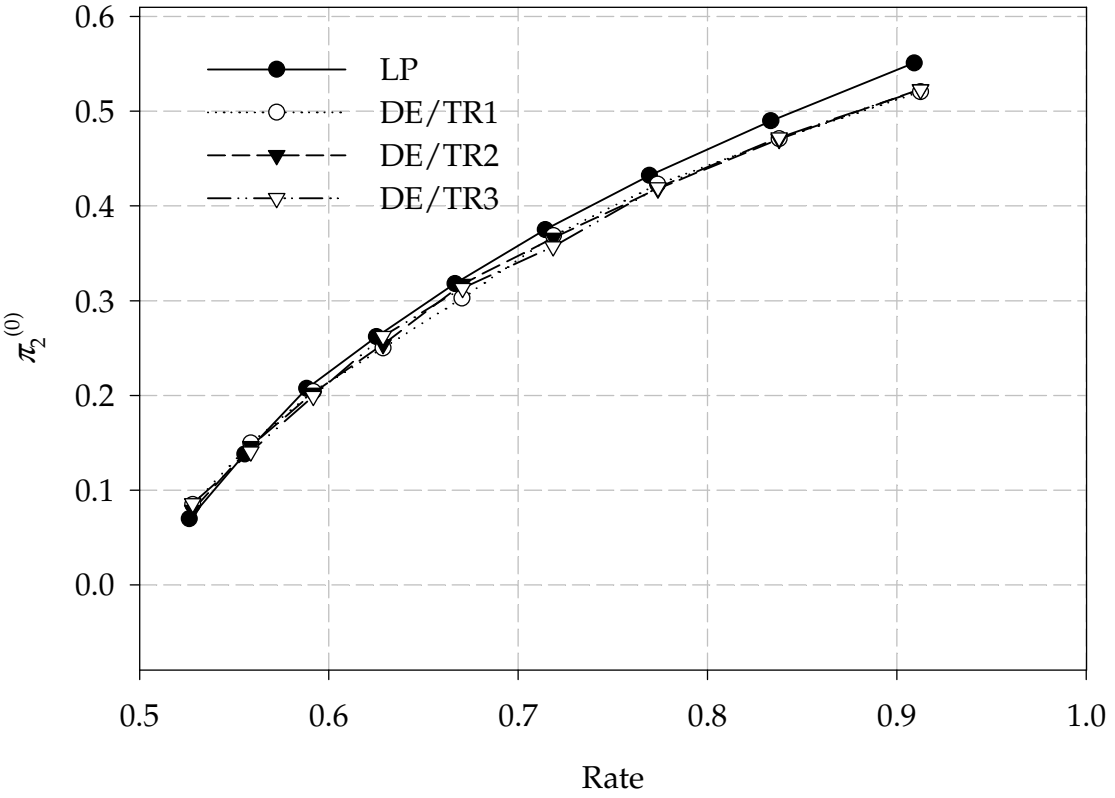


Figure 4.12: $\pi_2^{(0)}$'s designed with linear programming and DE over $\mathbb{U}^4$ (TR1) and DE with initial values from linear programming perturbed by uniform (TR2) and Gaussian (TR3) for a degree distribution pair, $\lambda(x) = 0.25105x + 0.30938x^2 + 0.00104x^3 + 0.43853x^9$ and $\rho(x) = 0.63676x^6 + 0.36324x^7$.

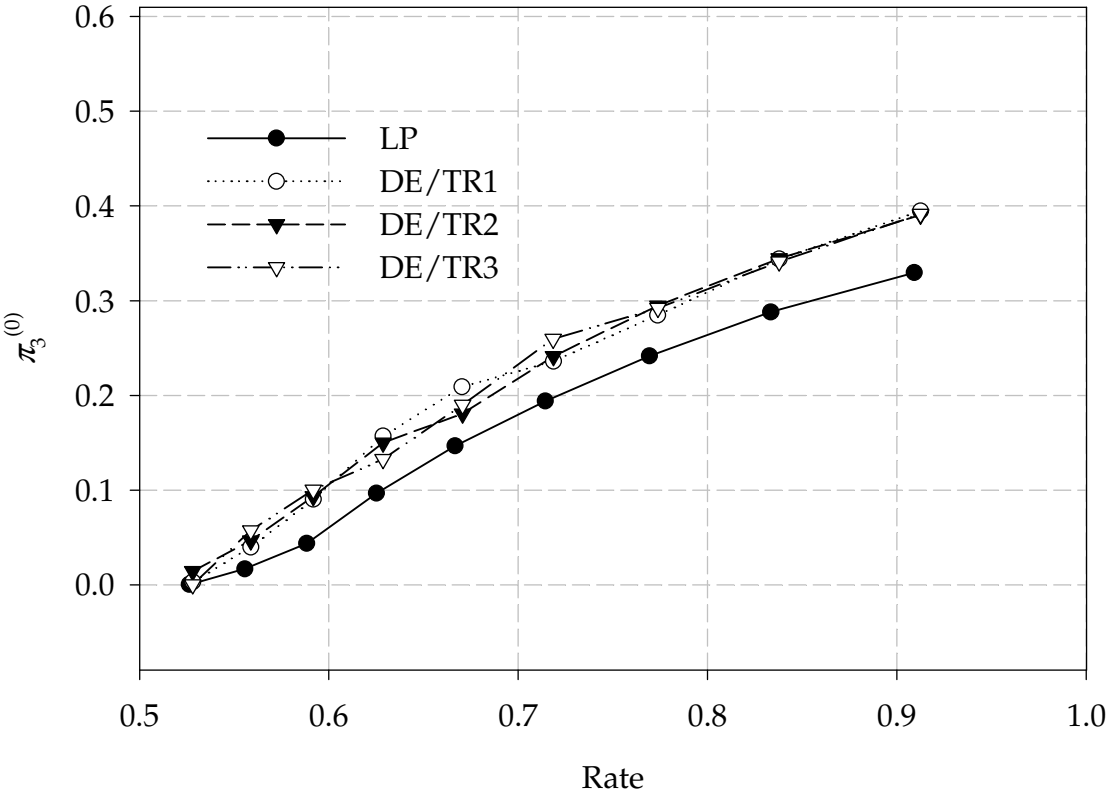


Figure 4.13: $\pi_3^{(0)}$'s designed with linear programming and DE over $\mathbb{U}^4$ (TR1) and DE with initial values from linear programming perturbed by uniform (TR2) and Gaussian (TR3) for a degree distribution pair, $\lambda(x) = 0.25105x + 0.30938x^2 + 0.00104x^3 + 0.43853x^9$ and $\rho(x) = 0.63676x^6 + 0.36324x^7$.

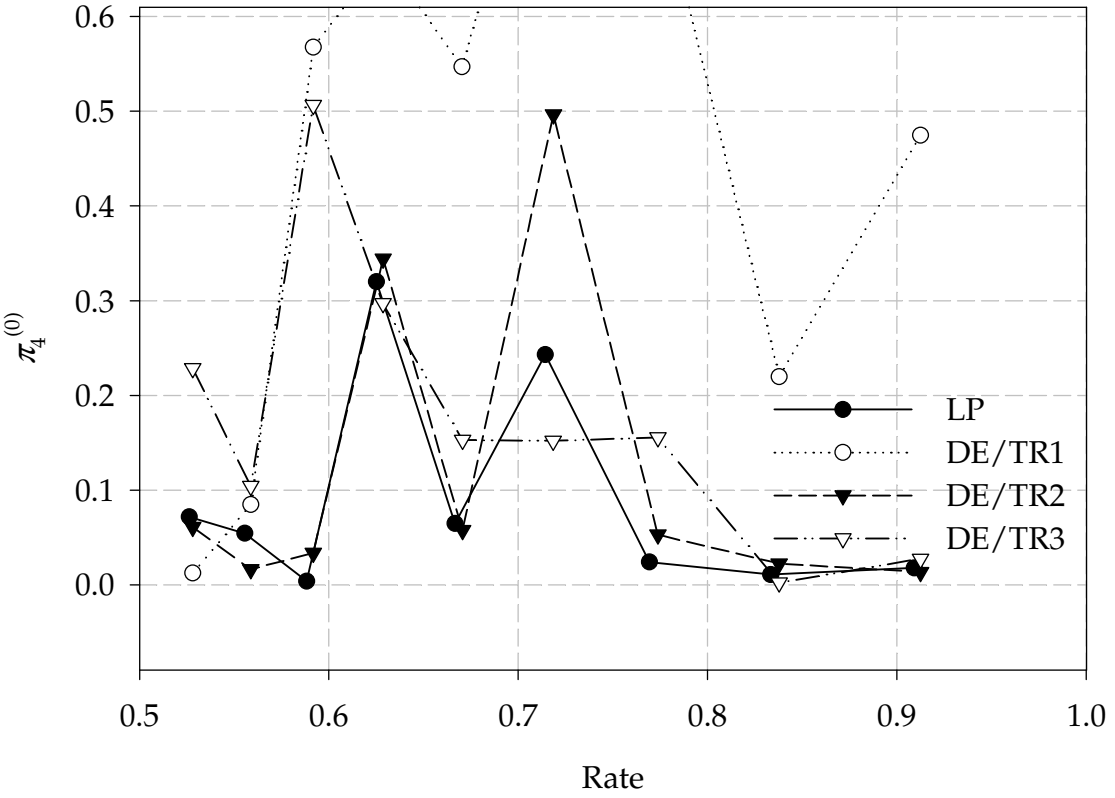


Figure 4.14: $\pi_4^{(0)}$'s designed with linear programming and DE over $\mathbb{U}^4$ (TR1) and DE with initial values from linear programming perturbed by uniform (TR2) and Gaussian (TR3) for a degree distribution pair, $\lambda(x) = 0.25105x + 0.30938x^2 + 0.00104x^3 + 0.43853x^9$ and $\rho(x) = 0.63676x^6 + 0.36324x^7$.

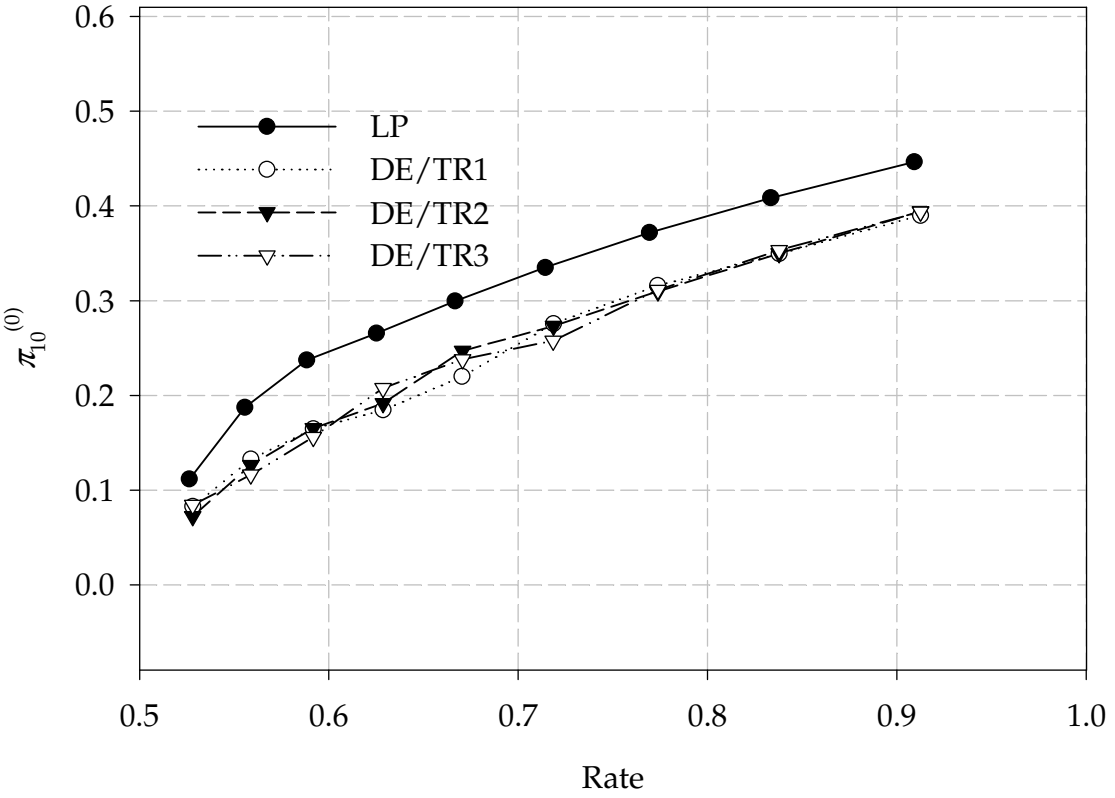


Figure 4.15: $\pi_{10}^{(0)}$'s designed with linear programming and DE over $\mathbb{U}^4$ (TR1) and DE with initial values from linear programming perturbed by uniform (TR2) and Gaussian (TR3) for a degree distribution pair, $\lambda(x) = 0.25105x + 0.30938x^2 + 0.00104x^3 + 0.43853x^9$ and $\rho(x) = 0.63676x^6 + 0.36324x^7$.

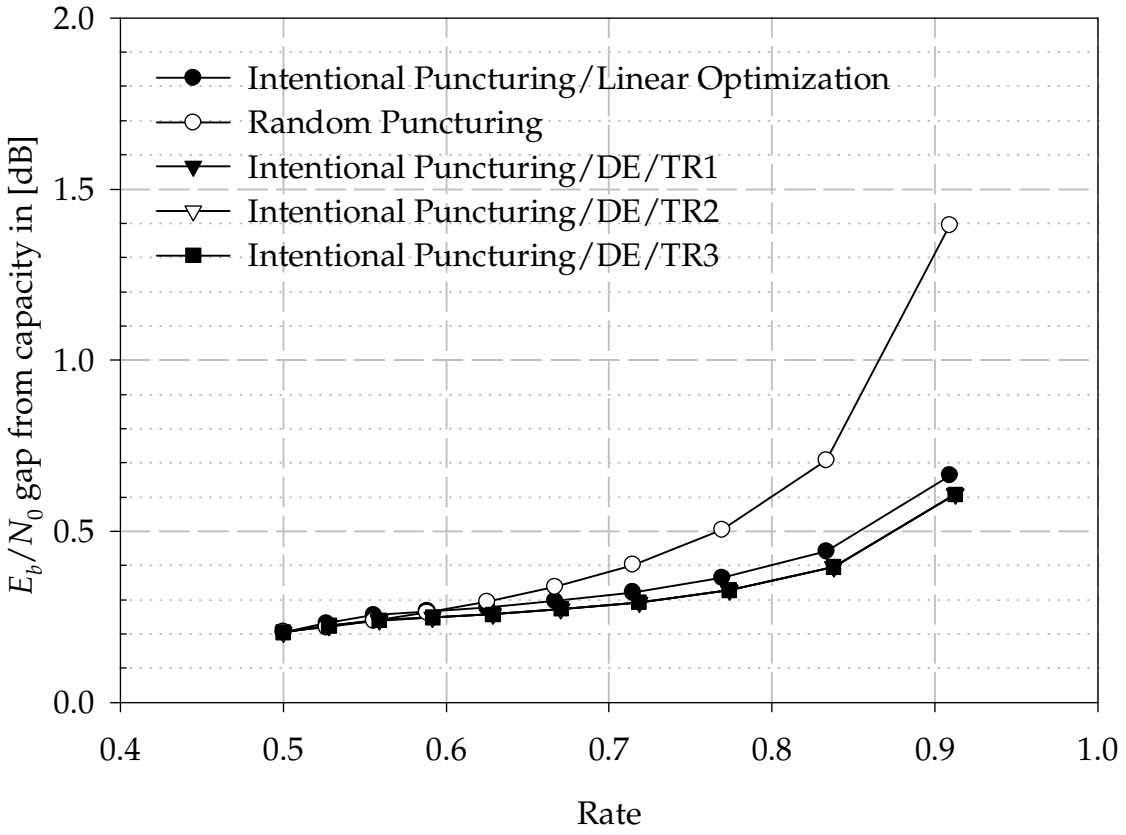


Figure 4.16: Gaps between asymptotic E_b/N_0's and the capacity of BPSK signal with respect to the coding rates for a degree distribution pair, $\lambda(x) = 0.25105x + 0.30938x^2 + 0.00104x^3 + 0.43853x^9$ and $\rho(x) = 0.63676x^6 + 0.36324x^7$.

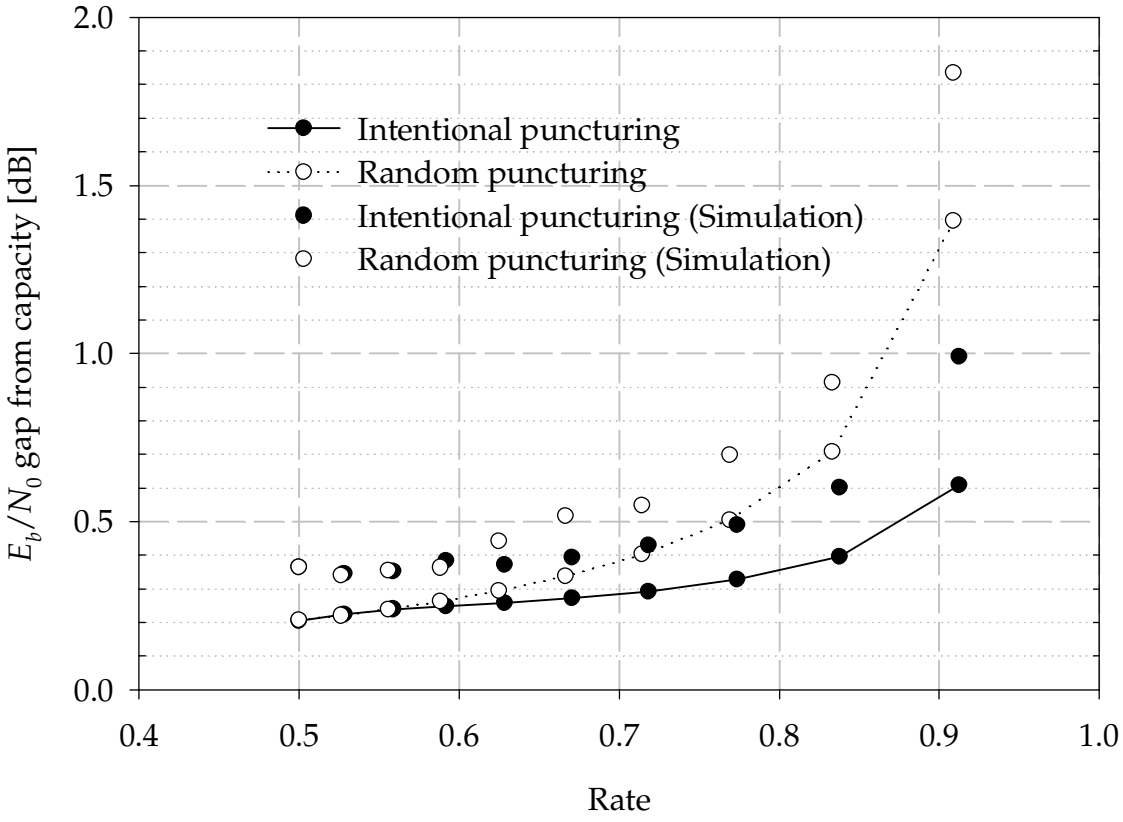


Figure 4.17: Gaps between asymptotic E_b/N_0's and the capacity of BPSK signal with respect to the coding rates for a degree distribution pair, $\lambda(x) = 0.25105x + 0.30938x^2 + 0.00104x^3 + 0.43853x^9$ and $\rho(x) = 0.63676x^6 + 0.36324x^7$.

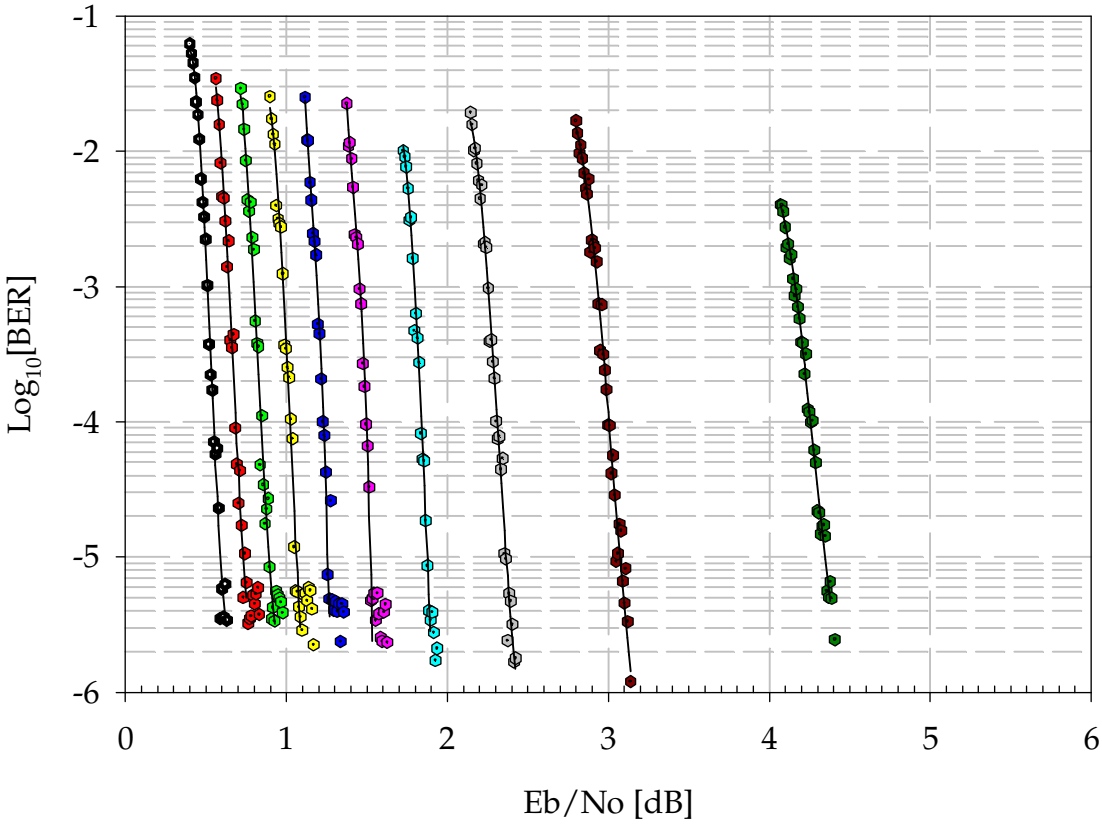


Figure 4.18: Bit-error rates of the LDPC code ($\lambda(x) = 0.25105x + 0.30938x^2 + 0.00104x^3 + 0.43853x^9$ and $\rho(x) = 0.63676x^6 + 0.36324x^7$) with the puncturing fractions, $p^{(0)} = 0.0000$, 0.05324, 0.10520, 0.15505, 0.20462, 0.25430, 0.30412, 035378, 0.40316, and 0.45194 (left to right).

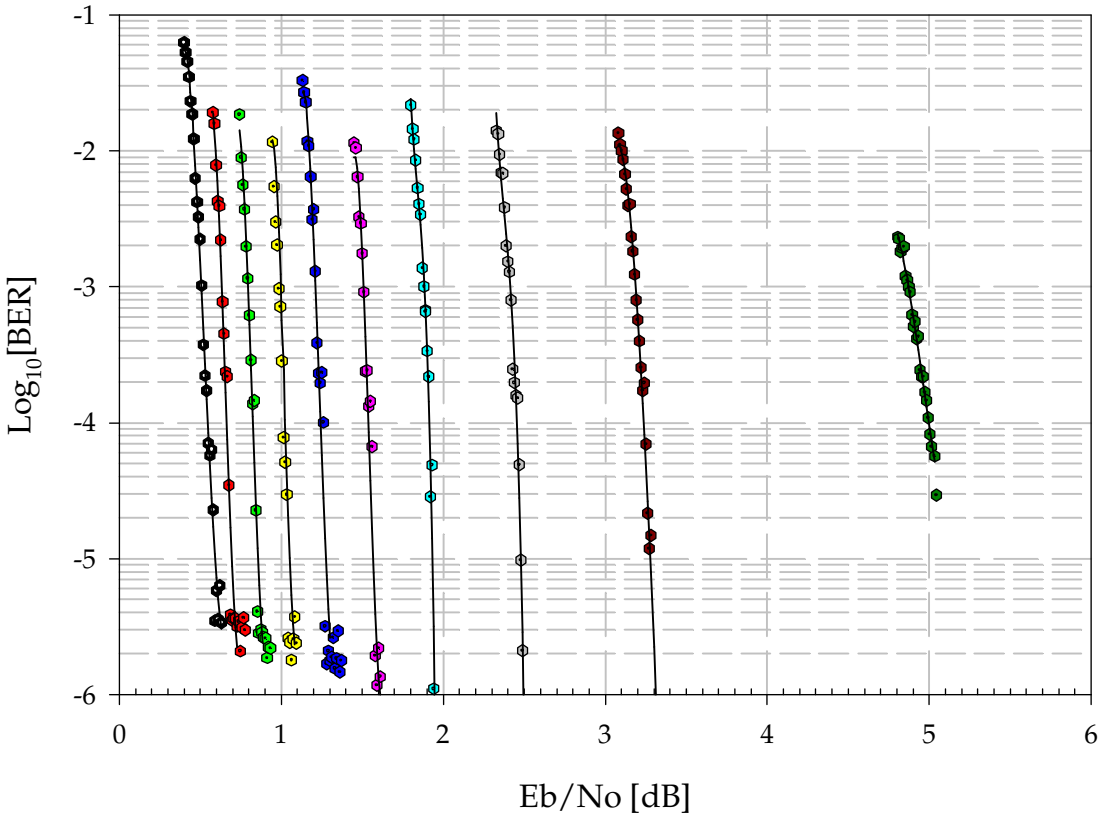


Figure 4.19: Bit-error rates of the LDPC code ($\lambda(x) = 0.25105x + 0.30938x^2 + 0.00104x^3 + 0.43853x^9$ and $\rho(x) = 0.63676x^6 + 0.36324x^7$) with the random puncturing fractions, $p^{(0)} = 0.00000$, 0.05000, 0.10000, 0.15000, 0.20000, 0.25000, 0.30000, 0.35000, 0.40000, and 0.45000 (left to right).

We design another puncturing distribution with a half rate LDPC code in [6] which has a degree distribution pair,

$$\lambda(x) = 0.23403x + 0.21242x^2 + 0.14690x^5 + 0.10284x^6$$
$$+0.30381x^{19},$$
and
$$\rho(x) = 0.71875x^7 + 0.28125x^8.$$

The puncturing distributions are obtained with linear programming and listed in Table 4.3. We also design the puncturing distributions with the combination of DE and DDE techniques. The puncturing probabilities for the symbols in G_2 are shown in Fig. 4.20 where the probabilities from LP and DE have a good match. The probabilities of the higher degrees have some discrepancies among results from LP and DE. Although we can have better puncturing distributions with DE, as we have explained, the improvement through DE is not distinctive. We evaluate the theoretical and simulated performances of the LDPC code with a code block length of 131072 and the results are depicted in Fig. 4.21. The simulated and theoretical results are consistent. The theoretical results with the puncturing distributions acquired through DE show that we can change the coding rate up to 0.86 at the expense of 0.1dB. In this experiment, the maximum coding rate is 0.95 that requires 0.3dB additional E_b/N_0 gap over that of the base code.

Table 4.3: Puncturing proportions, $\pi_j^{(0)}$'s for $\lambda(x) = 0.23403x + 0.21242x^2 + 0.14690x^5 + 0.10284x^6 + 0.30381x^{19}$ and $\rho(x) = 0.71875x^7 + 0.28125x^8$ at the overall puncturing probabilities of 0.00, 0.05, 0.10, 0.15, 0.20, 0.25, 0.30, 0.35, 0.40 and 0.45 with linear programming.

σ	0.96612	0.91177	0.88555	0.84154	0.79457	0.74385	0.68804	0.62472	0.54815	0.43501
$\pi_2^{(0)}$	0.00000	0.05264	0.10204	0.16435	0.22641	0.28807	0.34894	0.41032	0.46910	0.53375
$\pi_3^{(0)}$	0.00000	0.00867	0.06497	0.10882	0.14149	0.17505	0.21015	0.24330	0.28408	0.30992
$\pi_6^{(0)}$	0.00000	0.00028	0.06549	0.12196	0.21268	0.30055	0.38902	0.48388	0.56178	0.66375
$\pi_7^{(0)}$	0.00000	0.00029	0.00331	0.00011	0.00001	0.00021	0.00003	0.00004	0.00002	0.00001
$\pi_{20}^{(0)}$	0.00000	0.35051	0.39377	0.42155	0.44240	0.46617	0.48847	0.50541	0.53412	0.54837
$p^{(0)}$	0.00000	0.05000	0.10000	0.15000	0.20000	0.25000	0.30000	0.35000	0.40000	0.45000

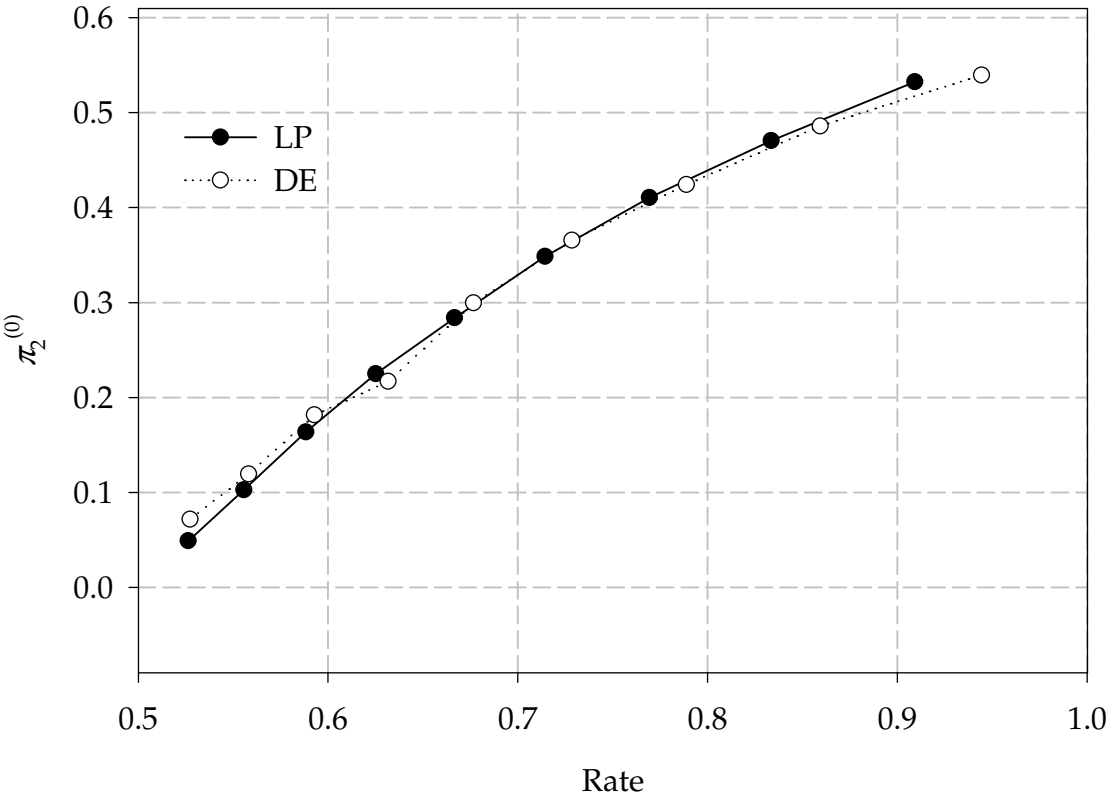


Figure 4.20: $\pi_2^{(0)}$'s designed with linear programming and DE over $\mathbb{U}^4$ (TR1) and DE with initial values from linear programming perturbed by uniform (TR2) and Gaussian (TR3) for a degree distribution pair, $\lambda(x) = 0.23403x + 0.21242x^2 + 0.14690x^5 + 0.10284x^6 + 0.30381x^{19}$ and $\rho(x) = 0.71875x^7 + 0.28125x^8$.

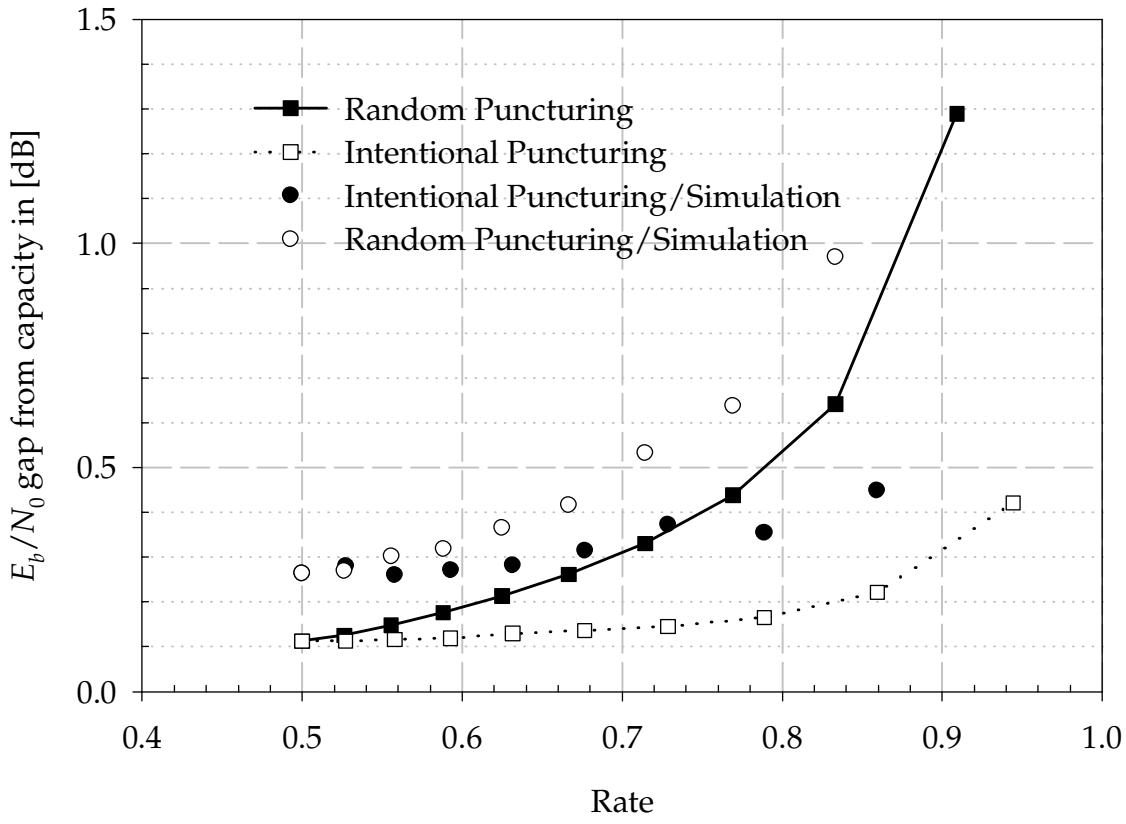


Figure 4.21: Gaps between asymptotic E_b/N_0's and the capacity of BPSK signal with respect to the coding rates for a degree distribution pair, $\lambda(x) = 0.23403x + 0.21242x^2 + 0.14690x^5 + 0.10284x^6 + 0.30381x^{19}$ and $\rho(x) = 0.71875x^7 + 0.28125x^8$.

4.4 Conclusions

We investigate good puncturing distributions for Rate-Compatible Punctured-Code (RCPC) with Low-Density Parity-Check (LDPC) codes. The theoretical performances of punctured LDPC codes are analyzed with Gaussian Approximation. The analytic results give us an insight into the convergence of punctured LDPC codes and enable us to predict the asymptotic performance of the punctured LDPC codes. From the analysis, we can see that the residual puncturing probabilities during iterations depend only on the structure of the punctured LDPC codes specified by a three-tuple degree distribution $\left(\lambda(x), \rho(x), \pi^{(0)}(x)\right)$ instead of the SNR, which simplifies the recursive expression of the updated mean, $m_u^{(k)}$ to the steady-state equation. The steady-state equation is a useful tool to visualize how a puncturing distribution works with a LDPC code and how we have to design a puncturing distribution to get a smaller E_b/N_0 threshold for a puncturing fraction, $p^{(0)}$ or a larger puncturing fraction for a given E_b/N_0.

Based on the analysis, we propose a design rule for good puncturing distributions, and the design rule is stated as a simple linear programming problem. Good puncturing distributions can be also designed with a combination of Discretized Density Evolution (DDE) and Differential Evolution (DE), and DDE models the message-passing decoders more accurately. However, DDE relies only on the numerical computations and cannot give us analytic insight. We verify the design rule based on GA by comparing E_b/N_0 thresholds of punctured LDPC codes designed with GA and DDE. The comparison shows there is less than 0.1dB discrepancy between results of two different design rules. The computational advantage of the proposed design rule can offset the difference. If more improved results are needed at the expense of the computational burden, we can use the results from the proposed design rule as initial values of the DE algorithm to speed up the design process.

We design puncturing distributions for LDPC codes in [6] and [37] which have the

maximum variable degrees of 10 and 20, respectively and evaluate the performances of the LDPC codes. The performance of the punctured LDPC code is measured by the gap between E_b/N_0 threshold and the capacity of BPSK with respect to coding rates. In the experiment with the first LDPC code, The gap of the base code is 0.2dB and an additional 0.1dB gap permits us to vary the coding rate up to 0.72. The changes of coding rate up to 0.84 and 0.92 are obtained at the expense of 0.2dB and 0.4dB additional gaps, respectively. In the experiment with the second LDPC code, the gap of the base code is 0.1dB, and we can change the coding rate up to 0.86 at the expense of 0.1dB. The maximum coding rate is 0.95 that requires 0.3dB additional E_b/N_0 to the gap of the base code. The LDPC codes are also implemented with code blocks of 131072, and the performances of the implemented codes are evaluated. We observe that the theoretical results are good lower bounds of the implemented LDPC codes.

Through the work done in this paper, we show the existence of good puncturing distributions for LDPC codes and propose a simple but effective design rule. This work is focusing mainly on theoretical aspects of the problem but can be further studied for designing practical rate-compatible punctured LDPC codes.

CHAPTER V

MORE ISSUES ON PUNCTURED LOW-DENSITY PARITY-CHECK CODES

5.1 Universality of Punctured Low-Density Parity-Check Codes in Excess Mutual Information Perspective

In Chapter 4, we showed that there exist good puncturing distributions for rate-compatible punctured Low-Density Parity-Check (LDPC) codes. The results give us a question about universality of LDPC codes over a broad range of coding rates. Jones *et al.* [24] [23] studied universality of LDPC codes over periodic fading channels. They claimed LDPC codes are universal in the Excess Mutual Information (EMI) sense instead of E_b/N_0 gap used in Chapter 4. They also claimed that E_b/N_0 gap is not a proper measure because for a given EMI, E_b/N_0 gap becomes wider at higher rates. We depict the Shannon limit of a BPSK signal and a relation between EMI and E_b/N_0, In Fig. 5.1 for the same EMI, E_b/N_0 gaps become wider (poorer) at higher data rates. The larger E_b/N_0 gap at higher data rates explains why the punctured LDPC codes of Chapter 4 have wider E_b/N_0 gaps at higher coding rates.

In the case that a code has an E_b/N_0 gap of 0.2dB at a coding rate of 0.5, the E_b/N_0 gap is equivalent to an EMI of 0.04265 [bits/channel use]. If the code is universal in the EMI sense over a range of coding rates (EMI is 0.04265 at each coding rate) by some means (including the puncturing scheme), the E_b/N_0 gap grows rapidly as shown in Fig. 5.2. This means that universality of LDPC codes in the EMI sense implies more losses of E_b/N_0 at higher rates.

In this section, we investigate universality of punctured LDPC codes in EMI sense over a broad range of coding rates (from 0.1 to 0.95). We evaluate performances of punctured LDPC codes with Discretized Density Evolution (DDE) introduced in Section 2.3 and design puncturing distributions with the combination of Differential Evolution (DE) and DDE as we did in Section 4.3.

We choose three base LDPC codes designed for coding rates of 0.1 from [25]. The edge degree distribution pairs and E_b/N_0 gaps of the codes are listed in Table 5.1. The code with a maximum left degree of 25 (called code 1) is chosen as a practical code because of its relatively small maximum left degree. The other two codes (called code 2 and code 3) are chosen for theoretical interest. Puncturing distributions for the three codes are designed, which change the coding rates from 0.1 (the base coding rate) up to 0.95. The puncturing distributions are listed in Tables 5.2, 5.3 and 5.4 for codes 1, 2, and 3 in Table 5.1, respectively. The designed puncturing distributions are optimized by DE, which is based on a random search algorithm. Thus, the distributions do not satisfy the rate-compatibility, $\pi_j^{(0)}(r_1) \leq \pi_j^{(0)}(r_2)$, where $r_0 \leq r_1 \leq r_2 \leq 1$, r_0 is a base coding rate, and r_1 and r_2 are coding rates of punctured LDPC codes. Although we can design the puncturing distributions in the rate-compatible fashion with more elaboration, we are only interested in the theoretical variations of EMIs and E_b/N_0 gaps.

We evaluate EMIs and E_b/N_0 gaps of the punctured LDPC codes and depict the results of the code 1, 2 and 3 in Figs. 5.3, 5.4, and 5.5, respectively. The results show that the punctured LDPC codes lose the optimality of the base code in the E_b/N_0 gap sense as the coding rate increases. However, the EMIs are invariant over the entire range of the coding rates. To see a better comparison, we depict the Shannon limit of a BPSK signal and the results of code 3 together in Fig. 5.6. The comparison gives us a conclusion that we can design universal LDPC codes over a range of coding rates with the puncturing scheme proposed in Chapter 4.

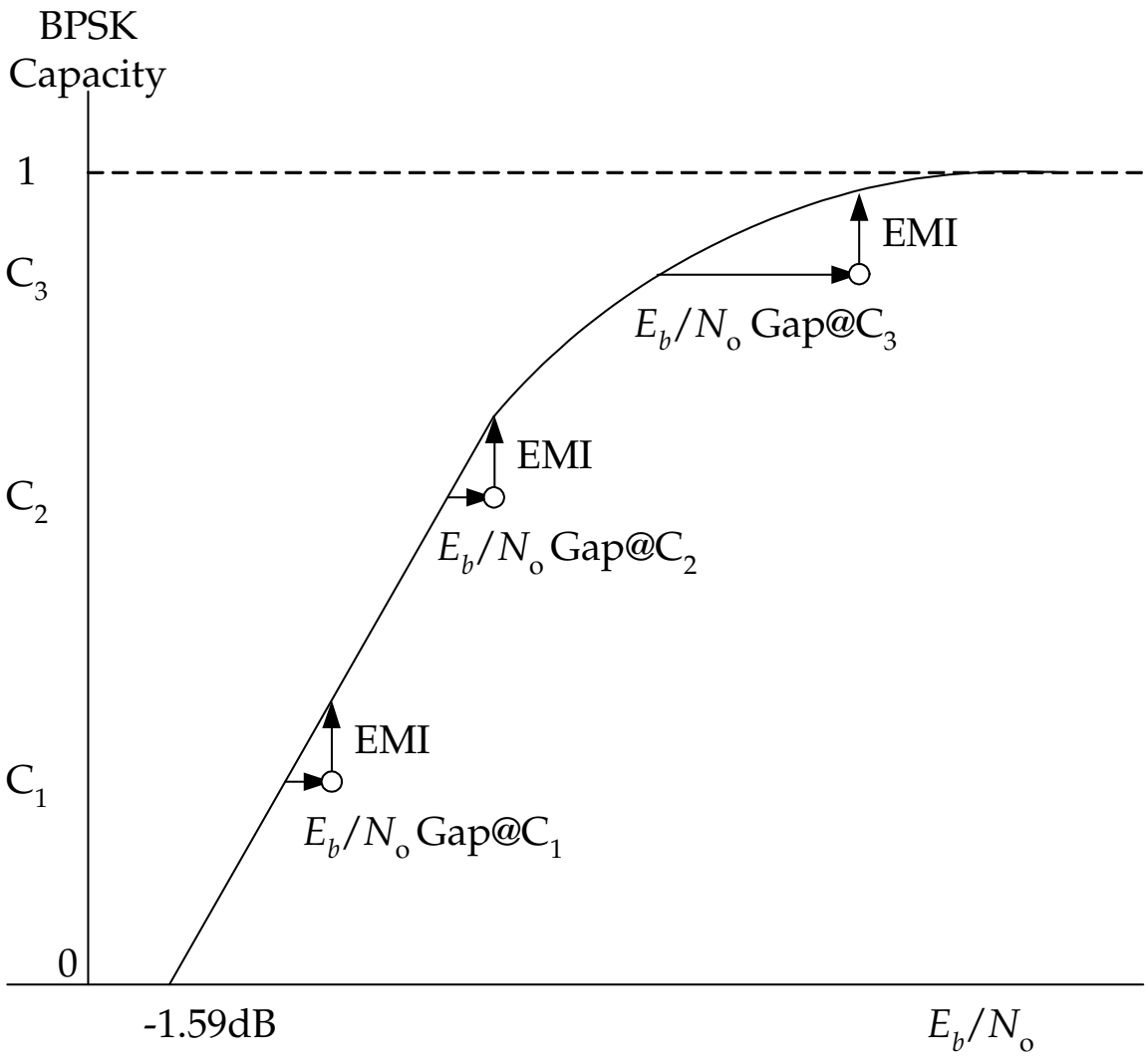


Figure 5.1: EMI versus E_b/N_0 gaps on the Shannon limit of BPSK signal.

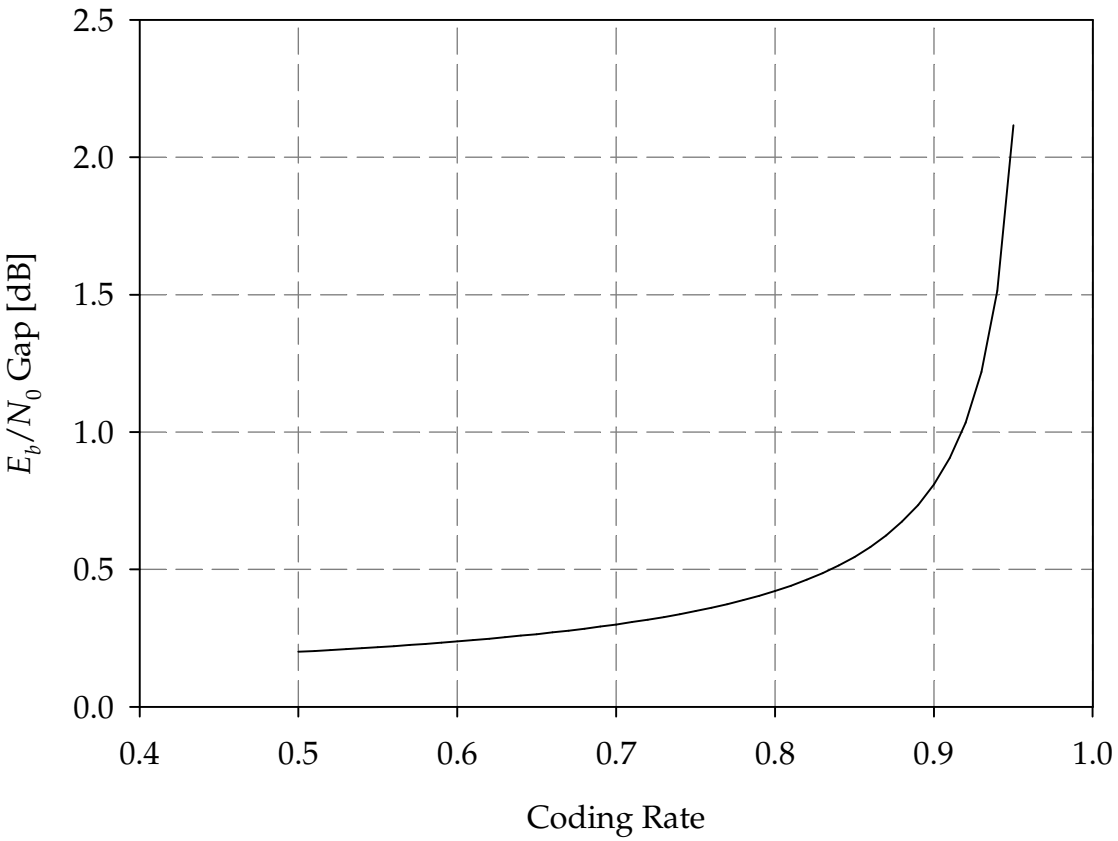


Figure 5.2: E_b/N_0 gaps for a uniform EMI (0.0.04265[bits/channel use]) with respect to coding rates.

Table 5.1: Edge degree distribution pairs of three rate 0.1 LDPC codes.

	Code 1		Code 2		Code 3	
i	λ_i	ρ_i	λ_i	ρ_i	λ_i	ρ_i
2	0.41493600		0.35926000		0.34770900	
3	0.18349200	0.4	0.16576700		0.15709400	
4	0.01300200	0.6		1.0	0.04095120	0.9
5	0.09308100		0.04667590		0.01181010	0.1
6			0.12147500		0.02383410	
7			0.00846288		0.14436300	
8	0.14701700					
13			0.08733650			
14			0.01425710			
15					0.03187300	
18					0.00645150	
20			0.00423524			
22					0.10221710	
23			0.01134680			
25	0.14847200					
30			0.07139350			
36			0.01026770		0.00554980	
38					0.01486220	
100			0.09952160		0.11328500	
E_b/N_0 gap [dB]	0.25789825		0.19965825		0.19405825	
EMI gap	0.08057932		0.06272632		0.06099868	

Table 5.2: Puncturing proportions, $\pi_j^{(0)}$'s for the code 1 in Table 5.1.

σ	2.51692	1.70666	1.33907	1.11013	0.94547	0.81560	0.70525	0.60384	0.49822
$\pi_2^{(0)}$	0.00000	0.48649	0.65558	0.74569	0.83847	0.97932	0.89520	0.91096	0.90413
$\pi_3^{(0)}$	0.00000	0.69715	0.83201	0.87184	0.65105	0.46819	0.84401	0.91573	0.96192
$\pi_4^{(0)}$	0.00000	0.03287	0.48916	0.38179	0.04527	0.71050	0.98541	0.23288	0.35996
$\pi_5^{(0)}$	0.00000	0.04248	0.33917	0.48427	0.95233	0.59816	0.42518	0.40977	0.96980
$\pi_8^{(0)}$	0.00000	0.69048	0.63990	0.74655	0.74808	0.79485	0.92976	0.99811	0.31757
$\pi_{25}^{(0)}$	0.00000	0.45209	0.76947	0.79130	0.80845	0.05765	0.30225	0.15915	0.89250
$p^{(0)}$	0.00000	0.50774	0.67066	0.75189	0.79475	0.82654	0.84922	0.86615	0.87916

Table 5.3: Puncturing proportions, $\pi_j^{(0)}$'s for the code 2 in Table 5.1.

σ	2.53386	1.70666	1.33907	1.11013	0.94547	0.81560	0.70525	0.60384	0.49822
$\pi_2^{(0)}$	0.00000	0.48039	0.66085	0.75565	0.80307	0.84239	0.86366	0.86481	0.90379
$\pi_3^{(0)}$	0.00000	0.67224	0.78360	0.77508	0.86437	0.85533	0.86620	0.99524	0.83540
$\pi_5^{(0)}$	0.00000	0.90355	0.65467	0.90892	0.33930	0.33440	0.93220	0.31218	0.89025
$\pi_6^{(0)}$	0.00000	0.24786	0.55064	0.64362	0.90315	0.96530	0.76840	0.89581	0.96477
$\pi_7^{(0)}$	0.00000	0.07257	0.40780	0.32610	0.63440	0.86378	0.82665	0.41150	0.33023
$\pi_{13}^{(0)}$	0.00000	0.32012	0.66570	0.99245	0.70607	0.90623	0.86689	0.96370	0.74835
$\pi_{14}^{(0)}$	0.00000	0.65242	0.06481	0.31989	0.79922	0.14357	0.63673	0.98773	0.90032
$\pi_{20}^{(0)}$	0.00000	0.95552	0.04241	0.32070	0.33155	0.33173	0.56204	0.74744	0.08820
$\pi_{23}^{(0)}$	0.00000	0.64426	0.56380	0.90536	0.34454	0.72550	0.57018	0.64116	0.99660
$\pi_{30}^{(0)}$	0.00000	0.87804	0.81818	0.16621	0.85512	0.86076	0.93026	0.97257	0.95968
$\pi_{36}^{(0)}$	0.00000	0.51439	0.88873	0.55390	0.60129	0.72442	0.39859	0.02069	0.73581
$\pi_{100}^{(0)}$	0.00000	0.12591	0.54387	0.96573	0.64446	0.39748	0.70320	0.34046	0.28174
$p^{(0)}$	0.00000	0.51364	0.67438	0.75420	0.80238	0.83378	0.85739	0.87301	0.88553

Table 5.4: Puncturing proportions, $\pi_j^{(0)}$'s for the code 3 in Table 5.1.

σ	2.53386	1.70666	1.33907	1.11013	0.94547	0.81560	0.70525	0.60384	0.49822
$\pi_2^{(0)}$	0.00000	0.49621	0.65102	0.76098	0.79763	0.83514	0.85734	0.88086	0.88271
$\pi_3^{(0)}$	0.00000	0.55677	0.87262	0.81008	0.94426	0.96328	0.98979	0.91127	0.98880
$\pi_5^{(0)}$	0.00000	0.83845	0.24233	0.22414	0.19179	0.13811	0.14649	0.72000	0.64585
$\pi_6^{(0)}$	0.00000	0.83722	0.67063	0.64384	0.46909	0.48380	0.61394	0.96941	0.49457
$\pi_7^{(0)}$	0.00000	0.94893	0.42185	0.97144	0.48651	0.67913	0.68656	0.26406	0.48412
$\pi_{13}^{(0)}$	0.00000	0.28332	0.66359	0.87782	0.93832	0.97364	0.98052	0.92635	0.92269
$\pi_{14}^{(0)}$	0.00000	0.94429	0.66862	0.23763	0.37460	0.68841	0.40356	0.93821	0.94674
$\pi_{20}^{(0)}$	0.00000	0.30900	0.27774	0.44806	0.64491	0.19529	0.99763	0.84213	0.11392
$\pi_{23}^{(0)}$	0.00000	0.22136	0.58738	0.61512	0.86735	0.73478	0.94901	0.82707	0.93375
$\pi_{30}^{(0)}$	0.00000	0.68436	0.75926	0.30178	0.30374	0.53229	0.30572	0.57428	0.58619
$\pi_{36}^{(0)}$	0.00000	0.84892	0.30383	0.91054	0.15455	0.63010	0.21184	0.96491	0.31919
$\pi_{100}^{(0)}$	0.00000	0.59937	0.70335	0.85659	0.77793	0.88603	0.77521	0.32784	0.73444
$p^{(0)}$	0.00000	0.51378	0.67440	0.75451	0.80272	0.83479	0.85744	0.87300	0.88682

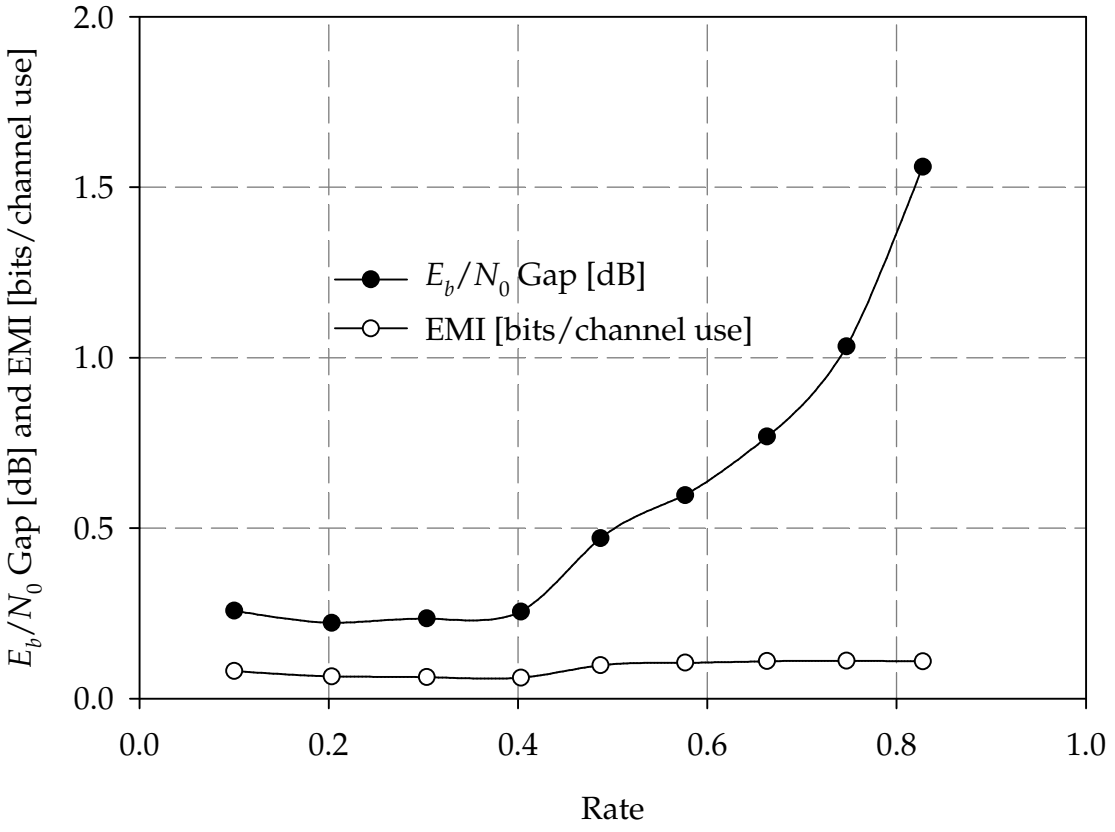


Figure 5.3: E_b/N_0 Gap and EMI changes with respect to coding rates (code 1 in Table 5.1).

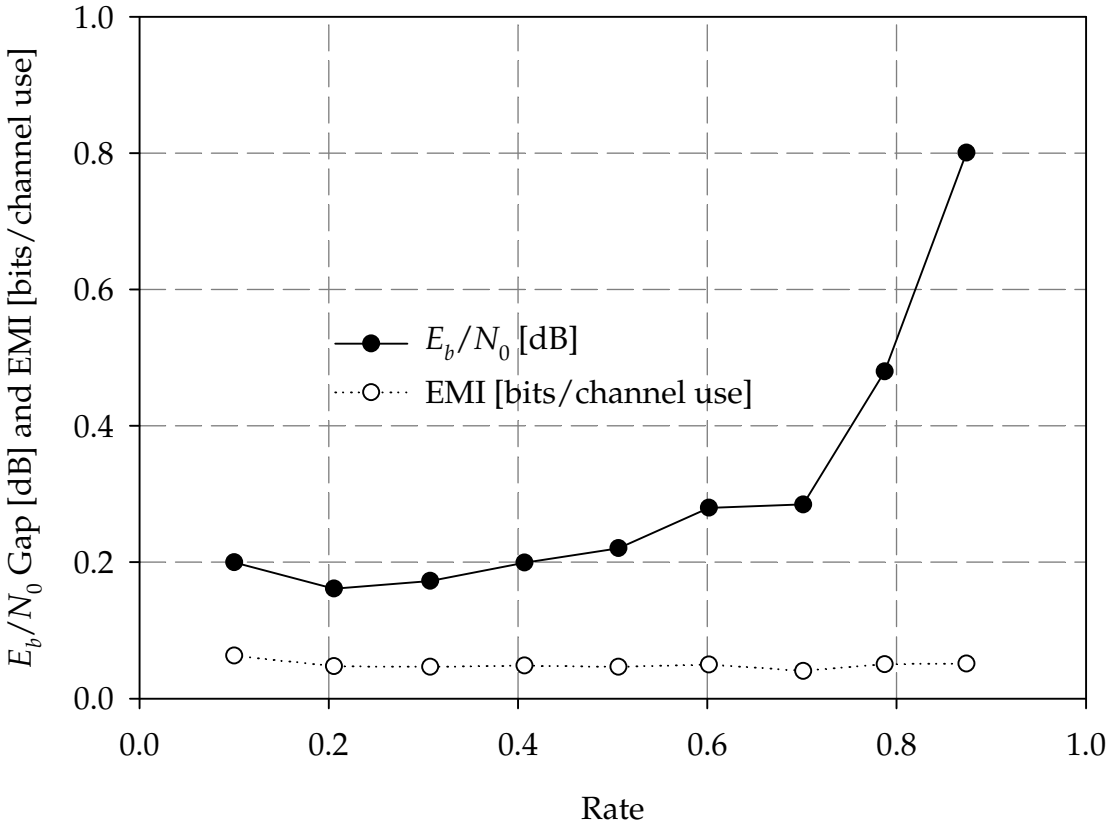


Figure 5.4: E_b/N_0 Gap and EMI changes with respect to coding rates (code 2 in Table 5.1).

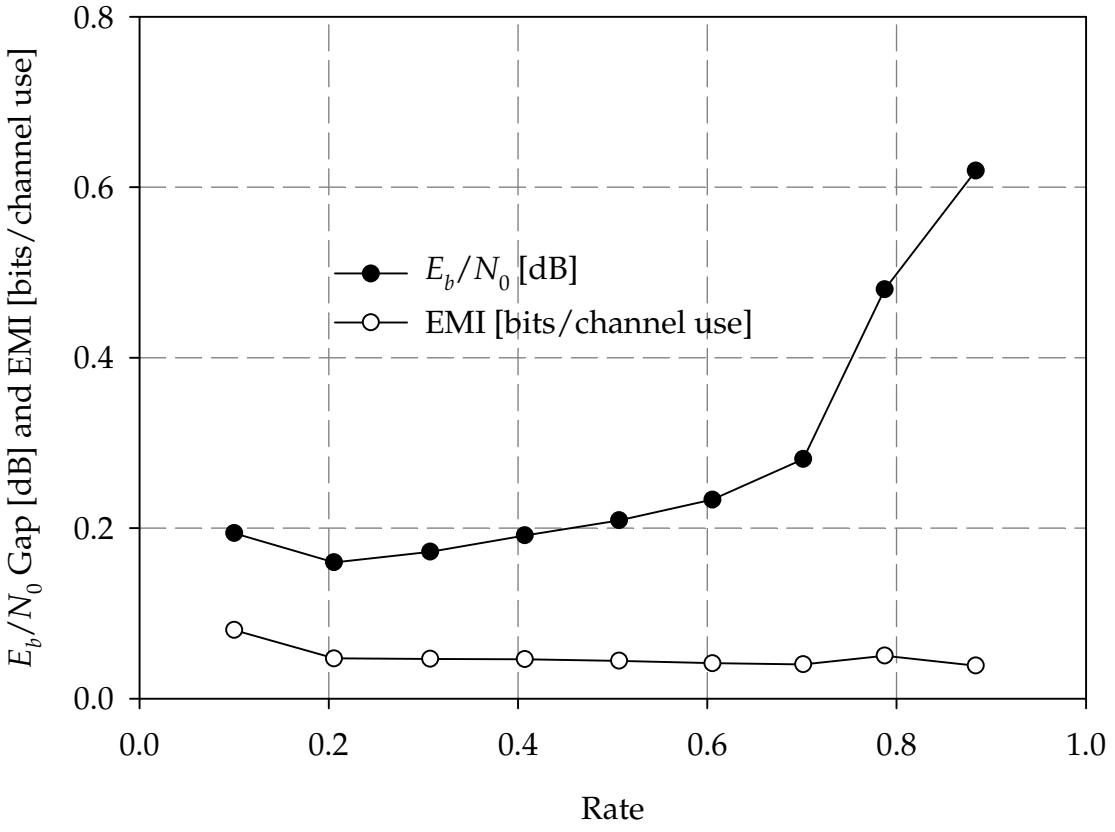


Figure 5.5: E_b/N_0 Gap and EMI changes with respect to coding rates (code 3 in Table 5.1).

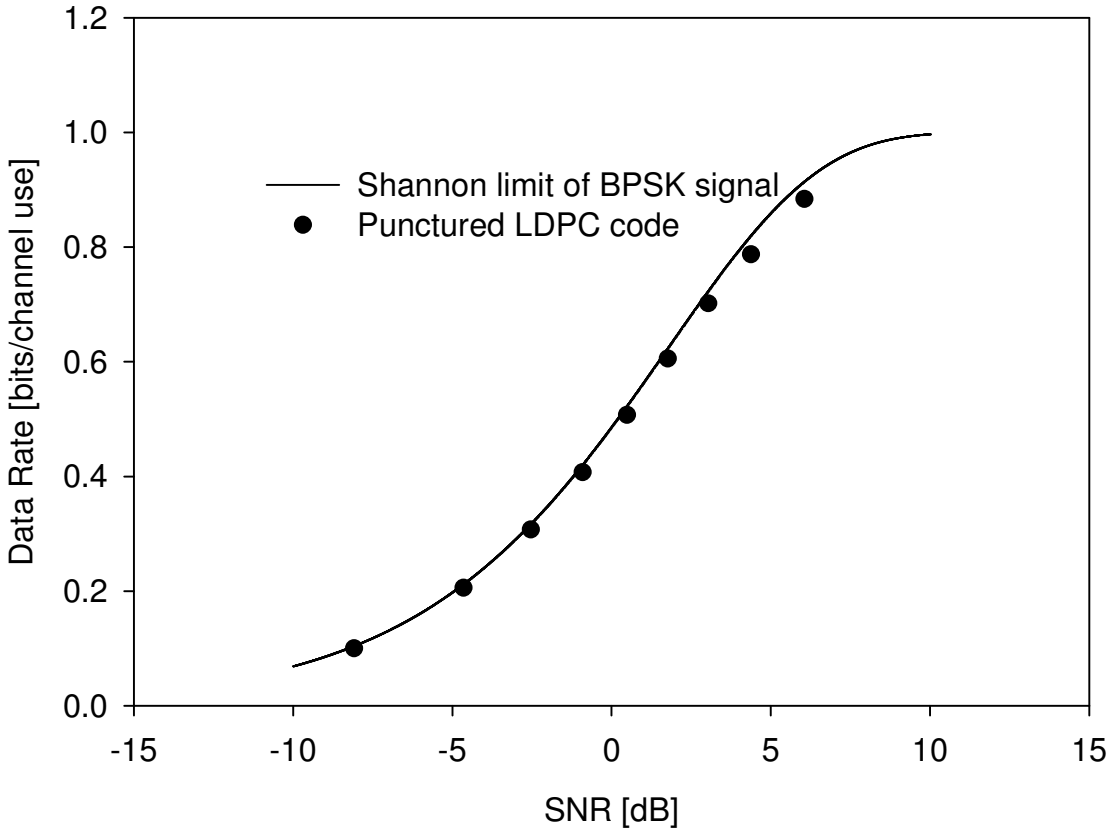


Figure 5.6: Comparison between the Shannon limit of BPSK signal and the EMI of the punctured LDPC code (code 3 in Table 5.1).

5.2 Design of High Rate Low-Density Parity-Check Codes with the Puncturing Scheme

The sum-product algorithm for decoding Low-Density Parity-Check (LDPC) codes converges to the optimal solutions when Tanner graphs of LDPC codes do not have cycles. Although cycle-free Tanner graphs satisfy the required condition for the optimality of the sum-product algorithm, cycle-free LDPC codes with finite block lengths (linear codes in general) have poor minimum distance [11]. Thus, we have to carefully design cycles in Tanner graphs to achieve better finite-length LDPC codes and smaller loss of decoding performance due to the cycles.

To the best of our knowledge, it is still an open problem to design cycles in LDPC codes. One well-known design criterion [33] [22] is to maximize a minimum cycle length (girth). We simply remove cycles of length 4, which prevents any two columns of a parity-check matrix from having more than one common non-zero term. That is, for a set of row indices

$$\theta_{j,k} = \{i | h_{i,j} = h_{i,k} \neq 0, 1 \leq i \leq r\},$$

the size of $\theta_{j,k}$ must be less than 2 ($|\theta_{j,k}| \leq 1$), where $1 \leq j \neq k \leq n$ and $h_{i,k}$ is the element on the ith row and the kth column of a parity-check matrix ($\mathbf{H}_{r \times n}$). Cycles of length 4 not only reduce the minimum distance of an LDPC code but also make a path through which a message from a node comes back to itself in two iterations. However, in designing high rate LDPC codes, it is difficult to remove cycles of length 4 because the parity-check matrices become dense as coding rates increase. Thus, high rate LDPC codes suffer from high error-floors that are due to small minimum distances.

In this section we propose a new way to design high rate LDPC codes with the puncturing scheme. In Section 5.1, we showed that punctured LDPC codes are universal in the EMI sense over a broad range of coding rates. Even with the E_b/N_0

gap measure, punctured LDPC codes have relatively small performance degradation. However, punctured LDPC codes have three advantages over LDPC codes designed for high rates, which is called hereafter *dedicated LDPC codes*; 1) Punctured LDPC codes have longer block lengths. That is, for a block length n, we puncture an LDPC code (base code) with a block length of $n/(1-p)$, where $0 \leq p < r_0$ is a puncturing fraction, and r_0 is a coding rate of the base code. Thus, we can take advantage of the longer block length as compared to the corresponding dedicated LDPC code. 2) We can avoid short cycle loops because we puncture a lower rate LDPC code which has a sparser parity-check matrix. In consequence of the second advantage, we can reduce error-floors of high rate LDPC codes. 3) Punctured LDPC codes are more amendable to type-II hybrid-ARQ protocol, which will be explained shortly.

We design an LDPC code with a coding rate of 0.8 with Gaussian approximation (GA). We also design another LPDC code for a coding rate of 0.8 by puncturing the LDPC code in Section 4.2 with the puncturing distributions in Table 4.2. We need a puncturing fraction (p) of 0.375 to change the coding rate from 0.5 to $0.8 = 0.5/(1-0.37500)$. Puncturing proportions ($\pi_j^{(0)}$'s) for the puncturing fraction (p) of 0.37500 is computed by linearly interpolating $\pi_j^{(0)}$'s for $p = 0.35378$ and 0.40316 in Table 4.2.

The edge degree distributions of the dedicated LDPC code and the punctured LDPC code with the puncturing proportions are listed in Table 5.5. The maximum left degree (10) is chosen to be practical but the maximum right degrees (8 and 20) are determined by the coding rates (0.5 and 0.8, respectively). The maximum right degree of the dedicated LDPC code becomes as high as 20, which is much higher than that of the punctured LDPC code. Due to the high maximum right degree, the dedicated LDPC code has a denser parity-check matrix which may cause unavoidable 4 cycle loops. The dedicated and punctured LDPC codes are implemented with block lengths of 2000 and 4000 which are practical block lengths. For the punctured LDPC

codes, the base codes have block lengths of 32000 (for 2000) and 6400 (for 4000) which are $1.6(= 1/(1 - 0.37500))$ times longer than the dedicated codes. We do our best to remove cycles of length 4 in parity-check matrices. We successfully remove 4 cycle loops in the parity-check matrices of the punctured LPDC codes but are not able to do so for the dedicated LDPC codes.

Table 5.5: Edge degree distribution pairs for rate 0.5 and 0.8 LDPC codes.

	$R = 0.5$			$R = 0.8$	
i	λ_i	ρ_i	$\pi_j^{(0)}$	λ_i	ρ_i
2	0.25105		0.44088	0.20996	
3	0.30938		0.31604	0.22211	
4	0.00104		0.03959	0.09455	
7		0.63676			
8		0.36324			
10	0.43853		0.32703	0.47338	
20					1.00000

A codeword of an LDPC code consists of message and parity parts, and we can arbitrarily allocate a portion of the codeword to the message part and the rest of the codeword to the parity part. Usually, variable nodes with higher degree are assigned to the message part because the variable nodes have more incoming messages and converge faster during iterations [37, 39].

In designing the dedicated codes, we assign higher degree variable nodes to the message part. However, in a codeword of the punctured LDPC codes, some high degree variable nodes are already punctured, which must be regarded as parity bits. Thus, we assign higher degree variable nodes less the punctured bits to the message part. Bit-error rates (BERs) of each LDPC code are evaluated with two different measures. The first BER measure counts errors only in the message parts of codewords and the other counts errors in both message and parity parts. As block length increases, the BER measures must have the same results, which is proved in Proposition 1. However, in finite-length LDPC codes, message bits with higher degrees have better BER performance than those of parity bits.

We evaluate BER performance of the dedicated and puntured LDPC codes and compare the results in Figs. 5.7 and 5.8. In the simulations, we observed 50 error codewords at each E_b/N_0 to get statistically reliable results. From the results, we notice the dedicated codes have better performances at low E_b/N_0 regions but have high error-floors due to short cycles as we expected. On the contrary, the punctured LDPC codes are poor at low E_b/N_0 regions but are better at high E_b/N_0 regions. Thus, we conclude that dedicated codes are suitable for error-tolerant service such as voice and video communications. For applications requiring low-error rates, punctured LDPC codes are more favorable due to the lower error-floors.

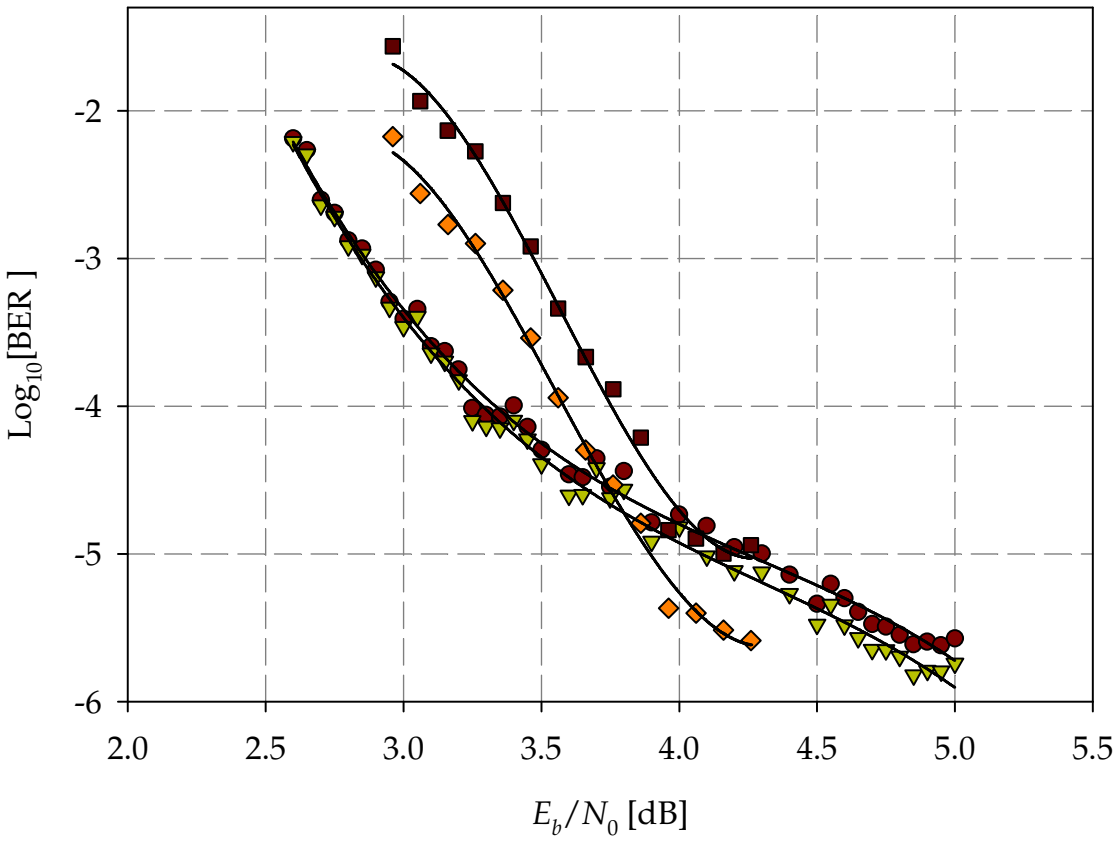


Figure 5.7: BER performances of dedicated and punctured LDPC codes with a block length of 2000, the circle and triangular symbols are BERs of the message part and the entire dedicated LDPC code, respectively and the square and diamond symbols are BERs of the message part and the entire punctured LDPC code, respectively.

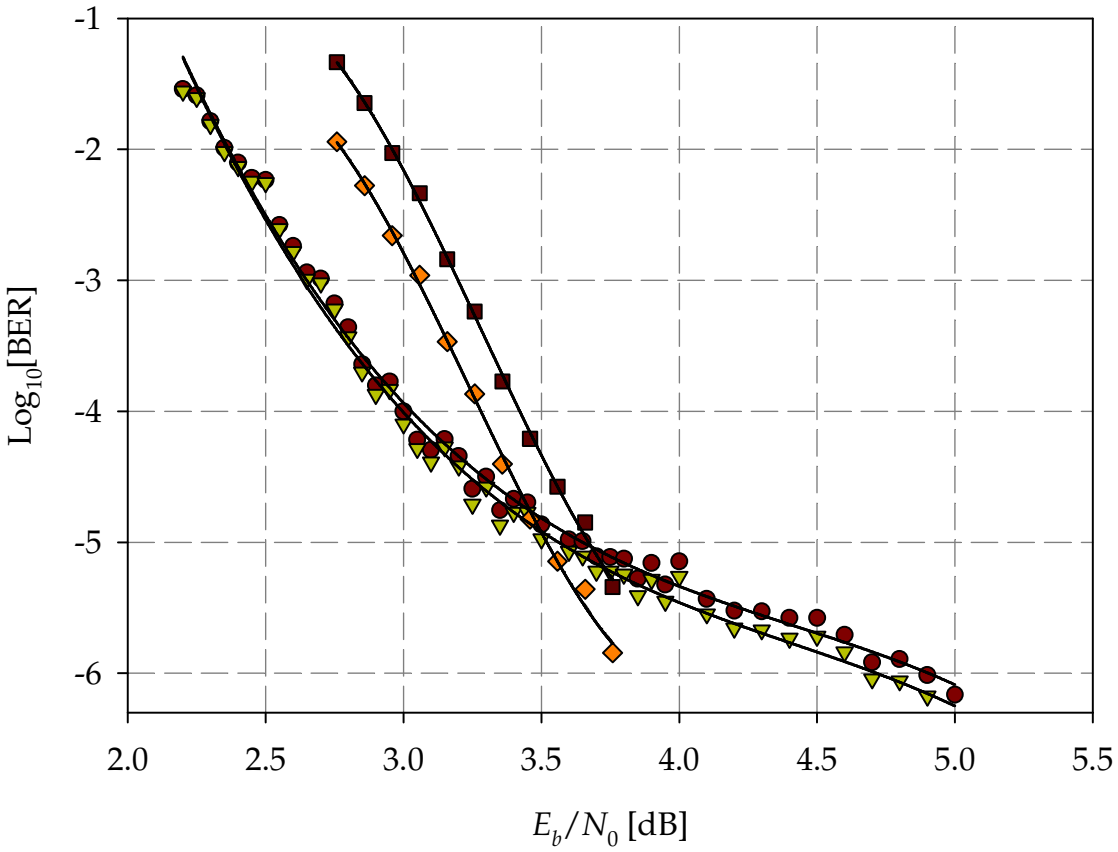


Figure 5.8: BER performances of dedicated and punctured LDPC codes with a block length of 4000, the circle and triangular symbols are BERs of the message part and the entire dedicated LDPC code, respectively and the square and diamond symbols are BERs of the message part and the entire punctured LDPC code, respectively.

For reliable communications, forward error correcting (FEC) with automatic repeat request (ARQ) protocol is widely adopted, which are called type-I and -II hybrid-ARQ protocols [44]. In the type-II hybrid-ARQ protocol, redundancies are progressively transmitted. That is, first, the protocol transmits a high rate codeword. If the transmitted codeword causes an unrecoverable error, the receiver requests more redundancies which will be combined with the previous codeword in the receiver and make a lower rate codeword. Punctured LDPC codes are amenable to the type II hybrid protocol as explained in Chapter 4. However, we did not investigate properties of frame-error rates (FER, rates of unrecoverable codewords) and decoder-error rates (DERs, rates of undetectable error codewords), which are key parameters in the ARQ protocols.

Here, we evaluate DERs and FERs of the dedicated and punctured LDPC codes and compare the results in Figs. 5.9 and 5.10. In both block lengths of 2000 and 4000, the punctured LDPC codes have much better FER and DER performances. Especially, at low E_b/N_0 regions, we cannot find decoder-error event with the punctured LDPC codes but the dedicated LDPC codes have high DERs. The simulation results indicate that the punctured LDPC codes have not only lower error floors but also better DERs and FERs.

Another feature of the type-II hybrid-ARQ protocol is code combining [5, 43] in which progressively transmitted redundancies are combined with the previous codeword. As more redundancies are combined, a codeword of a punctured LDPC code comes closer to that of the base code which is the best in the rate-compatible code set. Thus, transmission of redundancies promises better coding gain with punctured LDPC codes as long as base codes are well-designed. However, dedicated LDPC codes are best only at their coding rates. Progressively transmitted redundancies must be simply added to the previous codeword, which effectively reduces the designed coding rate. The rate change deteriorates the performance of dedicated LDPC codes. Thus,

codeword combining in dedicated LDPC codes is poorer than in punctured LDPC codes.

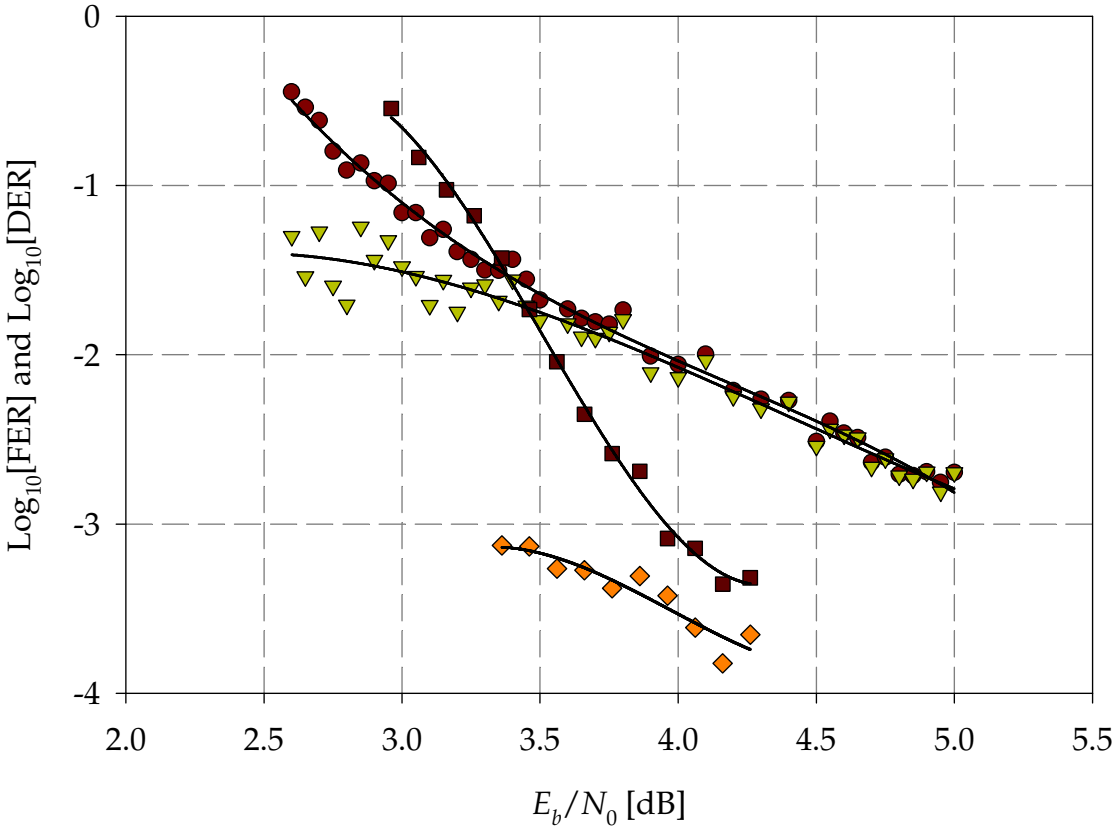


Figure 5.9: FER and DER performances of the dedicated and punctured LDPC codes with a block length of 2000, the circle and triangular symbols are FERs and DERs of the dedicated LDPC code, respectively and the square and diamond symbols are FERs and DERs of the punctured LDPC code, respectively.

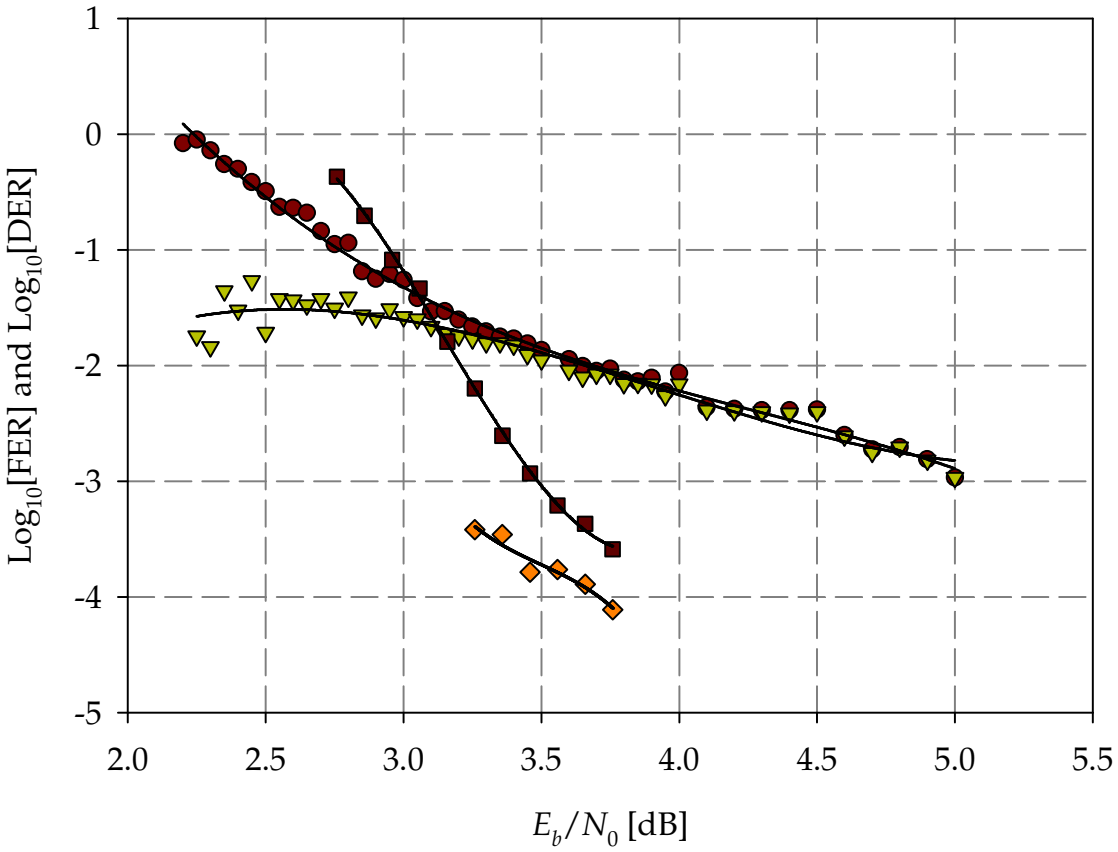


Figure 5.10: FER and DER performances of the dedicated and punctured LDPC codes with a block length of 4000, the circle and triangular symbols are FERs and DERs of the dedicated LDPC code, respectively and the square and diamond symbols are FERs and DERs of the punctured LDPC code, respectively.

5.3 Conclusions

In this chapter, we studied universality of punctured LDPC codes in the excess mutual information (EMI) sense. Although the E_b/N_0 gap may be a more favorable measure in some applications, the EMI measure is theoretically interesting because it is independent of coding rate. We evaluate thresholds of three punctured LDPC codes whose lowest coding rates are 0.1's. The puncturing distributions are designed to change the coding rates from 0.1 to 0.95. Across the broad range of coding rates, the punctured LDPC codes show uniform EMI values, which indicates the universality of punctured LDPC codes in the EMI sense. However, E_b/N_0 gaps become wider (poorer) at higher coding rates, which is predicted by the relation between EMI and E_b/N_0 gap on the capacity curve.

We also study behavior of finite-length punctured LDPC codes at a high rate (0.8 in our simulations) and compare punctured LDPC codes with LDPC codes designed for the coding rate. Punctured LDPC code are at most as good as the dedicated LDPC codes because the puncturing disturbs the optimality of base codes. However, punctured LDPC codes have three advantages over dedicated LDPC codes;

- Punctured LDPC codes have long effective block lengths,
- Short cycles in parity-check matrices of punctured LDPC codes can be easily removed,
- Punctured LDPC codes are more amenable to type-II hybrid-ARQ protocol thanks to an efficient code structure for code combining.

We implement two dedicated LDPC codes for a coding rate of 0.8 with block lengths of 2000 and 4000 that are compared with punctured LDPC codes. The punctured LDPC codes are implemented by puncturing 1/2 rate LDPC codes with block lengths of 3200 and 6400. In this simulations, the block lengths of the punctured

LDPC codes are 1.6 times longer than those of the dedicated LDPC codes. Besides the longer block lengths, the punctured LDPC codes have no 4 cycle loops in their parity-check matrices. However, we cannot completely remove 4 cycle loops in the parity-check matrices of the dedicated LDPC codes because the parity-check matrices are much denser that those of the punctured LDPC codes. Due to the unavoidable 4 cycle loops in the dedicated LDPC codes, the dedicated LDPC codes have high error-floors. Although the punctured LDPC codes are outperformed by the dedidcated LDPC coded at low E_b/N_0 regions, the punctured LDPC codes have lower error-floors.

The type-II hybrid-ARQ protocol needs good error-detection, error-correction capabilities and an efficient structure for code combining. Because decoder-error rates (DERs) and frame-error rates (FERs) account for error-detection and error-correction capabilities, respectively, we compare FERs and DERs of the punctured LDPC codes with those of the dedicated LDPC codes. Simulation results show that the punctured LDPC codes outperform the dedicated LDPC codes. Especially, at low E_b/N_0 regions, where the punctured LDPC codes have poorer BERs, the DERs of the punctured LDPC codes are almost undetectable. Besides better DERs and FERs, by replacing punctured symbols with progressively received coded symbols, we can efficiently realize code combining. The combining process makes a codeword approach that of the base LDPC code that has optimal performance. Thus, we can conclude that punctured LDPC codes are more amendable to type-II hybrid-ARQ protocol.

CHAPTER VI

REMARKS

6.1 Contributions

- *Analysis of Low-Density Parity-Check Codes over Gaussian Channels with Erasures*

  The mixture channel proposed in this thesis is practically interesting but there has not been any previous work with low-density parity-check (LDPC) codes. We analyze behaviors of LDPC codes based on Gaussian approximation. The analysis results give us a tool to predict thresholds of LDPC codes over the mixed channel.

- *A new way to design LDPC codes over the mixed channel*

  Although well-designed LDPC codes are still good over the mixed channel, LDPC codes have poorer performances at high erasure probabilities. We propose a design method for robust LDPC codes over the mixed channel. LDPC codes designed with the proposed method have uniform performances over a range of erasure probabilities, which is achieved at the minor sacrifice of performances at low erasure probabilities.

- *Evaluations of LDPC codes with practical considerations over the mixed channel*

  Finite-length LDPC codes are extensively evaluated with combinations of two different types of erasures (block and random erasures) and different values of maximum number of iterations. The result must be helpful to design practical LDPC codes over the mixed channel.

- *Rate Compatible Punctured Low-Density Parity-Check Codes*

We propose a new way to puncture LDPC codes for rate-compatibility. The proposed puncturing method minimizes performance losses due to puncturing across a broad range of coding rates. We also extend edge degree distribution pairs to three-tuple distributions that concisely describe punctured LDPC codes.

- *Analysis of punctured LDPC codes*

  Punctured LDPC codes are analyzed both in a purely numerical way with discretized density evolution technique and mathematically with Gaussian approximation. The former allows us to predict accurate performances of punctured LDPC codes and the latter gives us a close form of density evolutions during iterations. Based on the analysis with Gaussian approximation, we propose a design rule with linear optimization techniques.

- *Universality of LDPC codes*

  Universality of channel codes over a broad range of coding rates is a theoretically interesting topic. We show that LDPC codes can be universal with the proposed puncturing method. The universality is measured in the excess-mutual information (EMI) sense over a range from 0.1 to 0.95 in our simulations.

- *High rate LDPC code design with the puncturing technique*

  We apply the puncturing method for designing high rate LDPC codes. The high rate LDPC codes with the puncturing method have longer effective block lengths, and larger minimum distances. We show that larger minimum distances of high rate LDPC codes with the puncturing method result in better error-floors than those of dedicated LDPC codes for high rates.

- *Punctured LDPC codes with type-II hybrid ARQ protocol*

  In type-II hybrid ARQ protocol, frame-error rate (FER) and decoder-error rate (DER) are more important than bit-error rate (BER). We empirically show that

punctured LDPC codes have better FER and DER. In addition to FER and DER, punctured LDPC codes have an efficient structure for code combining that is a key element in type-II hybrid ARQ protocol.

6.2 Future Work

- *Punctured regular LDPC codes*

  The proposed puncturing method determines variable nodes to be punctured based on their degrees. Thus, the proposed method is not applicable for regular LDPC codes because variable nodes of regular LDPC codes have the same degree. Although their capacity-approaching performance makes irregular LDPC codes theoretically more attractive, in some applications, regular LDPC codes are more favorable because of their good minimum distance properties and simplicity. We will study how to puncture finite-length regular LDPC codes.

- *Finite-length punctured LDPC codes*

  The proposed puncturing method is based on the fact that LDPC codes do not have short cycles, which is satisfied with infinite block lengths. In some practical applications, block lengths must be several thousand bits or less with which the proposed puncturing method may not be the best. We are trying to find a better way to puncture finite-length regular/irregular LDPC codes.

- *Analysis of high rate punctured LDPC codes*

  We show that high rate LDPC codes with the puncturing method have better error-floors. However, we do not mathematically prove that punctured LDPC codes have better minimum distances than those of dedicated LDPC codes. We will study minimum distances of punctured LDPC codes in order to verify our empirical observations.

REFERENCES

[1] Bartle, R. G., *The Elements of Real Analysis*. New York: John Wiley & Sons, Inc., 2nd ed., 1976.

[2] Berrou, C., Glavieux, A., and Thitimajshima, P., "Near shannon limit error-correcting coding and decoding," in *Proc. IEEE Int. Conf. Commun.*, (Geneva, Switzerland), pp. 1064–1070, May 1993.

[3] Blake, I. F., *An introdcution to Applied Probability*. New York: John Wiley & Sons, 1976.

[4] Cain, J. B., G. C. Clark, J., and Geist, J. M., "Punctured convolutional codes of rate $(n-1)/n$ and simplified maximum likelihood decoding," *IEEE Trans. Inform. Theory*, vol. 25, pp. 97–100, Jan. 1979.

[5] Chase, D., "A maximum-likelihood decoding approach for combining an arbitrary number of noisy packets," *IEEE Trans. Commun.*, vol. COM-33, pp. 385–393, May 1985.

[6] Chung, S.-Y., *On the Construction of Some Capacity-Approaching Coding Schemes*. Ph.D. dissertation, Mass. Inst. Technol., Cambridge, MA, 2000.

[7] Chung, S.-Y., Forney, G. D., Jr., Richardson, T. J., and Urbanke, R. L., "On the design of low-density parity-check codes within 0.0045 dB of the shannon limit," *Electron. Lett.*, vol. 5, pp. 58–60, Feb. 2001.

[8] Chung, S.-Y., Richardson, T. J., and Urbanke, R. L., "Analysis of sum-product decoding of low-density parity-check codes using a Gaussian approximation," *IEEE Trans. Inform. Theory*, vol. IT-47, pp. 657–670, Feb. 2001.

[9] Clark, J. and Holton, D. A., *A First Look at Graph Theory*. 687 Hartwell Street, Teaneck, NJ 07666: World Scientific Publishing Co. Pte. Ltd., 1991.

[10] Di, C., Proietti, D., Telatar, I. E., Richardson, T. J., and Urbanke, R. L., "Finite-length analysis of low-density parity-check codes on the binary erasure channel," *IEEE Trans. Inform. Theory*, vol. IT-48, pp. 1570–1579, June 2002.

[11] Etzion, T., Trachtenberg, A., and Vardy, A., "Which codes have cycle-free Tanner graph," *IEEE Trans. Inform. Theory*, vol. IT-45, pp. 2173–2181, Sept. 1999.

[12] Fossorier, M. P. C., "Iterative reliability-based decoding of low-density parity check codes," *IEEE J. Select. Areas Commun.*, vol. 19, pp. 908–917, May 2001.

[13] Fossorier, M. P. C., Mihaljević, M., and Imai, H., "Reduced complexity iterative decoding of low-density parity check codes based on belief propagation," *IEEE Trans. Commun.*, vol. 47, pp. 673–680, May 1999.

[14] Gallager, R. G., "Low-density parity check coes," *IRE Trans. Inform. Theory*, vol. IT-8, pp. 21–22, Jan. 1962.

[15] Gallager, R. G., *Low-Density Parity Check Codes*. Cambridge, MA: MIT Press, 1963.

[16] Ha, J. and McLaughlin, S. W., "Analysis and design of LDPCCs over Gaussian channels with erasures," in *Proc. Int. Symp. Information Theory*, (Lausanne, Switzerland), p. 30, 2002.

[17] Ha, J. and McLaughlin, S. W., "Optimal puncturing distributions for rate-compatible low-density parity-check codes," *IEEE Trans. Inform. Theory*, 2002, submitted.

[18] Ha, J. and McLaughlin, S. W., "Low-density parity-check codes over Gaussian channels with erasures," *IEEE Trans. Inform. Theory*, vol. IT-49, pp. 1801–1809, July 2003.

[19] Ha, J. and McLaughlin, S. W., "Optimal puncturing distributions for rate-compatible low-density parity-check codes," in *Proc. Int. Symp. Information Theory*, (Yokohama, Japan), p. 233, 2003.

[20] Ha, J. and McLaughlin, S. W., "Optimal puncturing of irregular low-density parity-check codes," in *IEEE Int. Conf. Commun.*, (Anchorage, Alaska), pp. 3110–3114, May 2003.

[21] Hagenauer, J., "Rate-compatible punctured convolutional codes (RCPC codes) and their applications," *IEEE Trans. Commun.*, vol. COM-36, pp. 389–400, Apr. 1988.

[22] Hu, X., Eleftheriou, E., and Arnold, D. M., "Progressive edge-growth Tanner graphs," in *Proc. IEEE GLOBECOM*, (San Antonio, Texas), pp. 995–1001, Nov. 2001.

[23] Jones, C., Matache, A., Tian, T., Villasenor, J., and Wesel, R., "LDPC codes as universal space-time codes," in *to be published*.

[24] Jones, C., Tian, T., Matacbe, A., Wesel, R., and Villasenor, J., "Robustness of LDPC codes on periodic fading channels," in *Proc. IEEE GLOBECOM*, (Taipei, Taiwan), pp. 1284–1288, Nov. 2002.

[25] LTHC, Communications Theory Lab. Available: http://lthcwww.epfl.ch/research/ldpcopt/.

[26] Luby, M., Mitzenmacher, M., Shokrollahi, A., and Spielman, D., "Analysis of low density codes and improved designs using irregular graphs," in *Proc. 30th Annu. ACM Symp. Theory of Computing*, pp. 249–258, 1998.

[27] Luby, M., Mitzenmacher, M., Shokrollahi, A., Spielman, D., and Stemann, V., "Practical loss-resilient codes," in *Proc. 29th Annu. ACM Symp. Theory of Computing*, pp. 150–159, 1997.

[28] Lugand, L. R., Jr., D. J. C., and Deng, R. H., "Parity retransmission hybrid arq using rate 1/2 convolutional codes on a nonstationary channel," *IEEE Trans. Commun.*, vol. COM-37, pp. 755 – 765, July 1989.

[29] MacKay, D. J. C., "Good error-correcting codes based on very sparse matrices," *IEEE Trans. Inform. Theory*, vol. IT-45, pp. 399–431, Mar. 1999.

[30] MacKay, D. J. C. and Neal, R. M., "Near shannon limit performance of low-density parity-check codes," *Electron. Lett.*, vol. 32, pp. 1645–1646, Aug. 1996.

[31] Mandelbaum, D. M., "An adaptive feed-back coding using incremental redundancy," *IEEE Trans. Inform. Theory*, vol. IT-20, pp. 388–389, May 1974.

[32] Mao, Y. and Banihashemi, A. H., "Decoding low-density parity-check codes with probabilistic scheduling," *IEEE Commun. Lett.*, vol. 5, pp. 414–416, Oct. 2001.

[33] Mao, Y. and Banihashemi, A. H., "A heuristic search for good low-density parity-check codes at short block lengths," in *IEEE Int. Conf. Commun.*, (Helsinki, Finland), pp. 41–44, June 2001.

[34] Pearl, J., *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. San Mateo: Morgan Kaufmann, 1988.

[35] Pishro-Nik, H., Rahnavard, N., Ha, J., and Fekri, F., "Low-density parity-check codes for volume holographic memory systems," *Appl. Opt.*, vol. 42, pp. 861–870, Feb. 2003.

[36] Price, K. and Storn, R., "Differential evolution-a simple and efficient heuristic for global optimization over continuous spaces," *J. Gobal optimiz.*, vol. 11, pp. 341–359, 1977.

[37] Richardson, T. J., Shokrollahi, A., and Urbanke, R. L., "Design of capacity-approaching irregular low-density parity-check codes," *IEEE Trans. Inform. Theory*, vol. IT-47, pp. 619–637, Feb. 2001.

[38] Richardson, T. J. and Urbanke, R. L., "The capacity of low-density parity-check codes under message-passing decoding," *IEEE Trans. Inform. Theory*, vol. IT-47, pp. 599–618, Feb. 2001.

[39] Richardson, T. J. and Urbanke, R. L., "Efficient encoding of low-density parity-check codes," *IEEE Trans. Inform. Theory*, vol. IT-47, pp. 638–656, Feb. 2001.

[40] Sipser, M. and Spielman, D. A., "Expander codes," *IEEE Trans. Inform. Theory*, vol. IT-42, pp. 1710–1722, Nov. 1996.

[41] Tanner, M., "A recursive approach to low complexity codes," *IEEE Trans. Inform. Theory*, vol. IT-27, pp. 533–547, Sept. 1981.

[42] Tian, T., Jones, C., Villasenor, J. D., and Wesel, R. D., "Construction of irregular LDPC codes with low error floors," in *IEEE Int. Conf. Commun.*, (Anchorage, Alaska), pp. 3125 –3129, May 2003.

[43] Wicker, S. B. and Bartz, M. D., "Type-II hybrid-ARQ protocols using punctured MDS codes," *IEEE Trans. Commun.*, vol. COM-42, pp. 1431–1440, Apr. 1994.

[44] Wicker, S. B., *Error Control Systems for Digital Communications and Storage*. Upper Saddle River, New Jersey 07458: Prentice Hall, 1995.

[45] Winston, W. L., *Introduction to Mathematical Programming: Applications and Algorithms*. Belmont, California: Duxbury Press, 2nd ed., 1995.

[46] Yang, M. and Ryan, W., "Design of LDPC codes for the burst-erasure channel with awgn," *to be published*.

VITA

Jeongseok Ha received the B.E. degree in electronics from Kyungpook National University, Taegu, Korea in 1992 and M.S. degree in Electrical and Computer Engineering from the Pohang University of Science and Technology, Pohang, Korea, in 1994. From 1994 to 1999, he was a research engineer with Electronics and Telecommunications Research Institute, Taejun, Korea where he was involved in developing base stations of IS-98 cellular system and wireless local loop. Since 1999, he has been working toward the doctoral degree with School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA. His research interests include digital communications and error-control systems.